Una plataforma móvil para la enseñanza de la literatura inglesa

TRABAJO DE FIN DE GRADO

Miguel Torres Porta

Departamento de Ingeniería del Software e Inteligencia Articial

Facultad de Informática

Universidad Complutense de Madrid

Junio 2016

Documento maquetado con TEXIS v.1.0+.

Este documento está preparado para ser imprimido a doble cara.

Una plataforma móvil para la enseñanza de la literatura inglesa

Ingeniería del Software e Inteligencia Artificial
IT/2016/6

Dirigida por el Doctor
Gonzalo Méndez Pozo

Departamento de Ingeniería del Software
e Inteligencia Articial
Facultad de Informática
Universidad Complutense de Madrid

Junio 2016

Il faut avoir déjà beaucoup appris de choses
pour savoir demander ce qon ne sait pas.
Jean-Jacques Rousseau

Agradecimientos

Primero de todo a mis padres y hermanos que han hecho esto posible.

A Gonzalo Mendez por haberme dado este proyecto, sabiendo los riesgos que conllevaba mi estancia en Francia durante todo el año y por haberme guiado y dedicado tiempo a lo largo de todo el proceso de desarrollo. A la Dra. Ana de la Universidad Autónoma de Madrid por haber confiado en nosotros este proyecto.

A todos mis amigos que durante todo este año me han ayudado y soportado.

Autorización

Se autoriza a la Universidad Complutense de Madrid (UCM) a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a su autor el Trabajo Fin de Grado (TF) cuyos datos se detallan a continuacion. Así mismo autorizo a la Universidad Complutense de Madrid a que sea depositado en acceso abierto en el repositorio institucional con el objeto de incrementar la difusion, uso e impacto del TFG en Internet y garantizar su preservacion y acceso a largo plazo.

Miguel Torres

Resumen

Actualmente vivimos en una epoca de cambios en la que podemos ver como las tecnologías están presentes en todos lo ámbitos de nuestra vida y cada día avanzan más rapido. Es por eso que el objetivo principal de este trabajo de fin de grado es una propuesta para la innovación educativa, en colaboración con la Universidad Autónoma de Madrid, de actualizar la educación a los tiempos que corren, de utilizar las tecnologías que tenemos a nuestra dispoción para facilitar tanto a profesores como a alumnos la labor de enseñanza y aprendizaje.

Este proyecto es de nueva creación y se propone la creación de una plataforma móvil que incorpore información creada por los propios alumnos, para que puedan, estudiar o repasar contenidos vistos en clase directamente en sus propios móviles. Para ello se van a utilizar herramientas de desarrollo de aplicaciones móviles, en nuestro caso Android Studio.

Como proposito se encuentra en crear una aplicación que se funcional en la mayoría de los dispositivos actuales, y que por lo tanto no requiera de unas altas especificaciones para su correcto funcionamiento, ni tenga requerimientos excesivos de memoria y batería.

Tambien se ha diseñado una plataforma web a través del cual los usuario incluyan su propia información.

Se propone estudiar también como los alumnos aceptan este tipo de propuesta.

Abstract

Nowadays we live in a time of change in which we can see how the technologies are presents in all areas of our lives and each day they move faster. It's because that, the main objetive of this final degree project is a proposal for teaching innovation, in collaboration with the Universidad Autónoma de Madrid, to update the education at the current times, to use the technologies that we have in our hand to make easier both professors and students the teaching and learning task.

This project is a new creation project and is propose the creation of a mobile platform who incorporates information created by the students, so that they can study or review content seen in class directly on their phones. To do this we are going to use mobile applications development tools, in our case Android Studio.

The main purpose is to create an application that is functional in most current devices, and therefore do not requiere high specifications for proper operations, of have excesive requests for memory and battery.

Design a web platform through which the user include their own information.

Also study as the students accept this type of proposal.

Palabras Clave

Actividades

Aplicación Android

Aprendizaje Literatura Inglesa

Plataforma de aprendizaje

Keywords

Activities

Android Application

Learning English Literature

Learning platform

Índice

Índice de figuras

Capítulo 1

Introducción

No es la más fuerte de las especies la que sobrevive, ni tampoco la más inteligente, si no la que mejor se adapta a los cambios.

Charles Darwin

1.1. Motivación

La influencia que las tecnologías móviles tienen en nuestras vidas aumenta cada día. Un buen uso de estas tecnologías puede facilitar nuestras vidas y ser la solución a muchos de nuestros problemas.

Nuestra sociedad comienza ser consciente del cambio que estamos sufriendo cada vez más notable y es por eso la necesidad de adaptar estos cambios a todos lo ambitos que nos rodean, estén o no relacionados con la tecnología y sobretodo en uno tan importante como la educación, que ha de avanzar, evolucionar y adaptarse a los cambios.

Tener al alcance herramientas y técnicas para facilitar la labor tanto a profesores como a alumnos es fundamental para adaptar la educación a los tiempos que corren. Es por eso que se realiza este proyecto de innovación educativa para la Universidad Autonoma de Madrid tutelado por la Dra. Ana González-Rivas Fernández por parte de la Universidad Autonoma de Madrid (UAM) y con la colaboración de alumnos de grado y máster en estudios ingleses para desarrollar una plataforma web y móvil que apoye la enseñanza de asignaturas relacionadas con la literatura inglesa a traves de la realización de ejercicios de tipo test elaborados por los propios alumnos de la UAM.

1.2. Objetivos

Los objetivos de este proyecto desarrollado junto con los alumnos de la Universidad Autonoma de Madrid (UAM) consisten en la realización de una plataforma a través de la cual los alumnos puedan incluir información de diferentes autores y épocas relacionados con la historia de la literatura inglesa y con un cuestionario para cada autor o época que facilite el aprendizaje de cada uno y su posterior evaluación de lo aprendido mediante la realización de los cuestionarios.

Para llevar acabo estos objetivos se han determinado los siguientes objetivos:

- El desarrollo de una aplicación móvil Android que permita, de una manera sencilla e intuitiva, por un lado el aprendizaje de los diferentes autores y por otro lado la realización de los test.
- Creación de una plataforma web mediante la cual los alumnos puedan incluir la información de los autores y sus respectivos cuestionarios realizado por ellos mismos, siendo ellos los encargados de poder actualizar los contenidos, sin necesidad de una persona con conocimientos avanzados de informática. La plataforma web debe ser lo suficientemente segura y estricta que no permita subir información erronea debido al desconocimiento de los usuarios y que pueda poner en riesgo la estabilidad y funcionamiento de toda la plataforma, solo aceptando documentos con una estructura y formato previamente definidos.

Este proyecto se llevará a cabo mediante los lenguajes de programación web HTML, CSS y PHP para la plataforma web y la herramienta AndroidStudio, para la realización de la aplicación, programada en Java y documentos XML.

Para la gestión de los test y de los usuarios es necesario el uso de bases de datos relacionales SQL.

1.3. Estructura del Documento

Como consiste en un proyecto de realización de una aplicación esta memoria se va a realizar de manera iterativa, es decir por cada versión que obtengamos de la aplicación de documentará como se ha realizado, que se ha utilizado en ella, cambios respecto a la version anterior y problemas encontrados.

Se realizará tambien una evalución con los usuarios y alumnos de la aplicación a fin de poder obtener *feedback* y opiniones de errores y aspectos a mejorar.

El documento consta de tres partes diferenciadas:

- La primera parte trata sobre introducción y trabajo realizado, aplicación y proyectos similares.
- La segunda parte se explica de manera detallada como se ha realizado la aplicación. En esta parte, la más extensa de dedicará a la explicación técnica del proyecto, metodologías y tecnologías que se han utilizado y que como se han utilizado durante todo el desarrollo. También incluirá una evalución por parte de los usuarios.
- En la tercera y ultima parte se habla sobre las conclusiones y el trabajo futuro

Capítulo 2

Introduction

2.1. Motivation

The influence that mobile technologies have in our lives increases every day. A good use of these technologies can facilitate our lives and can be the solution to lots of our problems.

Our society begins to be conscient about the change we're facing which is each time more evident. This is the reason to adjust these changes to all of the different fields that surround us, regardless of their relation with technology, and above all in education, which has to make a progress, evolve and adapt to these changes.

To adapt education to our times, it is fundamental to have tools and techniques in reach of professors and students to make easier their task. This is the reason behind this educative innovation project for the Universidad Autónoma de Madrid, supervised by the Dr. Ana González-Rivas Fernández from the same university, and with the collaboration of students in their career or master in English studies, with the objective of developing a web and mobile platform which helps the teaching of courses and subjets related with English literature, through the realization of multiple choice exercises, developed by students of the UAM themselves.

2.2. Objectives

The objectives of this project developed with students of the Universidad Autónoma de Madrid (UAM) consist in the fulfilment of a platform where

students can include information about different authors and periods related to the history of English literature, and a questionnaire for each author and period, and its subsequent evaluation of the learned through the questionnaires.

To bring out these objectives, the following objectives have been determined:

- The development of an Android mobile application which allows, simply and intuitively, the learning of different authors and the realization of their tests.
- The creation of a web platform where students can include information of the authors and their corresponding questionnaires produced by themselves, being able to update contents, without the necessity of someone with informatics knowledge. The web platform has to be sufficiently secure and strict that it does not allow mistaken information to be uploaded, due to the ignorance of users and to the threat towards the stability and the running of the platform. It will only accept documents with a structure and format previously defined.

This project will be accomplished through web programming languages such as HTML, CSS and PHP for the web platform. Java and XML documents will be used for the AndroidStudio tool.

The use of SQL relation date bases is necessary for the management of the tests.

2.3. Structure of the document

As the project consists in the implementation of an application, the report will be developed iteratively, this means that for each version of the application that we obtain, it will be noted down how it has been done, what has been used in it and all the changes that have been done regarding the previous version, as well as all the problems found.

In order to obtain some feedback and opinions of the errors and aspects that we could improve, an evaluation with the users and students of the application will be developed.

The document consists of three differentiated parts:

- The first part consists of an introduction and work done, the application and similar projects.
- In the second part the way the application has been build up is explained in detail. In this part, the largest part will be dedicated to the explanation of the technique used in the project, the methodologies and the technologies that have been used and how they have been

used during all of the process. An evaluation of the users will also be included.

- The third and last part includes the conclusions and the future projects.

Capítulo 3

Trabajo relacionado

Un artista copia, un gran artista roba.
Pablo Picasso

Aunque no hay aplicaciones que realicen exactamente lo mismo podemos encontrar multitud que realizan funciones parecidas.

Ya sea desde las aplicaciones clásicas para aprender idiomas mediante la realización continua de test a traves de los cuales la dificultad incrementa en función del nivel que vaya obteniendo el usuario a las aplicaciones en las que se deben resolver simples ejercicios ya sean matemáticos o de histoia, todas tienen un objetivo común, la de resolver ejercios con una finalidad educativa. Muchas de estas por no decir todas estan *Gamificadas*, es decir el preceso de realización de test está hecho como un juego para amenizar el proceso.

Como es complicado o no hay aplicaciones que cumplan los requisitos que necesitamos, al ser un proyecto de nueva creación vamos explicar la diferentes aplicaciones realcionadas de donde hemos sacado ideas para nuestro proyecto.

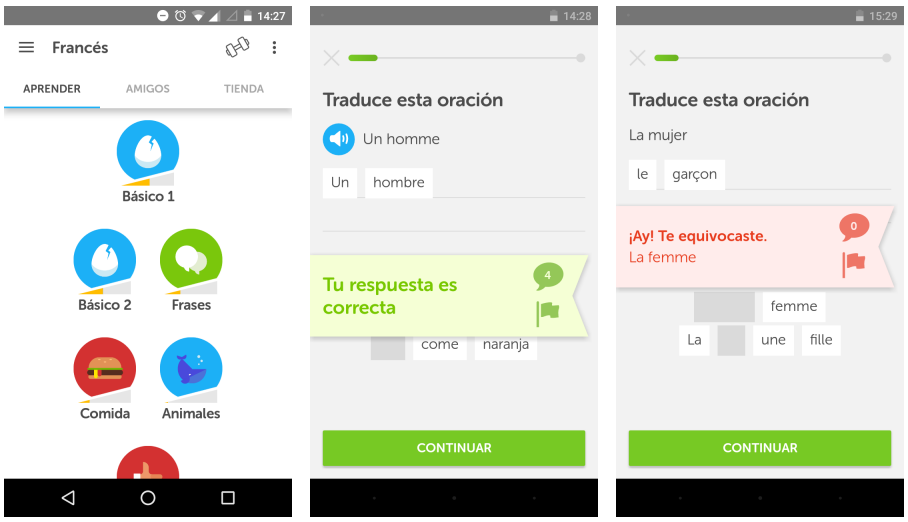
3.1. Duolingo

Duolingo es tanto una plataforma web como una aplicación móvil, nosotros únicamente nos hemos centrado en la aplicación.

Su objetivo es ofrecer cursos de idiomas gratuitos en la que su característica principal es la práctica a traves de la realización de test y preguntas de diferentes tipos. La dificultad de estas preguntas se incrementa a medida que el usuario va avanzando.

La forma de presentar los diferentes test es en forma de lista como lo podemos ver en la Figura 3.1 (a). Esto nos ha servido de ejemplo para nuestro proyecto a la hora de mostrar la lista de los test.

En cuanto a la realización de test hemos cogido alguna ideas como podemos ver en la Figura 3.1 (b) cuando el usuario responde correctamente el



(a) Lista de los test (b) Respuesta correcta (c) Respuesta incorrecta

Figura 3.1: Aplicación Duolingo.

sistema muestra un mensaje y espera a la confirmación del usuario. Cuando responde incorrectamente tambien muestra un mensaje Figura 3.1 (c), pero esta vez en color rojo y con la respuesta correcta.

Tambien hay tipos de preguntas que pueden sernos interesantes como el tipo *Multiple Choice* (Figura 2.3), que nosotros tendremos que implementar este tipo. Podemos ver también en la misma Figura que para algunos test la aplicación da al usuario vidas, esto nos puede ser de utilidad para el proceso de *Gamificacion* del que hemos hablado anteriormente.

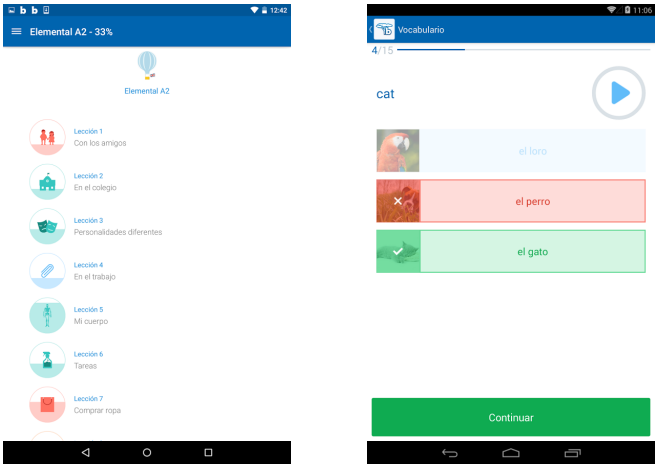
3.2. Busuu

Esta aplicación móvil tambien esta orientada a la enseñanza de idiomas.

En la figuta 3.2 (a) podemos ver la lista de los diferentes test, el diseño de este puede sernos de utilidad. Podemos observar en la Figura 3.2 (b) un ejercicio tipo de esta aplicación, son de remarcar el uso de colores y simbolos que pueden hacer al usuario tener una experiencia óptima a la hora de usar la aplicación.

3.3. Conclusiones

Entre estas dos aplicaciones explicadas hay muchas que realizan un trabajo similar de las que hemos sacado ideas y que hemos adaptado a nuestras necesidad para tener éxito en nuestro proyecto, es decir todo pasar por hacer



(a) Lista de los diferentes test

(b) Ejercicio tipo

Figura 3.2: Aplicación Busuu.

al usuario tener una buena experiencia de uso, facilitando la labor de aprendizaje y todo esto a traves de colores, diseños y animaciones atractivas, que por decirlo de alguna manera atrapen al usuario.

Capítulo 4

Primer prototipo

Subir montañas encrespadas requiere
pequeños pasos al comienzo.
William Shakespeare

Este prototipo puede ser considerado un mero esbozo del objetivo final de la aplicación ya que no era funcional. El objetivo principal era la de recoger los test creados por terceros a traves de plantillas y volcar dichos test a la aplicación a fin de poder realizar las diferentes cuestiones obteniendo su solución. Dichos tests tenían que tener una estructura fijada, que se explicará a continuación para así poder ser leido de manera correcta por la aplicación. En cuanto al elemento encargado de leer estos documentos, su funcionamento y su estructura será explicado tambien. En terminos de diseño, este distaba mucho de lo deseado pero era suficiente para hacerse una idea ya que el objetivo de este primer prototipo era poder procesar los test.

4.1. Diseño

En este primer prototipo el diseño era muy primitivo como se puede ver en la Figura 4.1, en la que solo había un único test, y al ábrirlo las diferentes preguntas salian en forma de lista.

Las interfaces estaban realizadas con documentos .xml. El main era una *Activity* del tipo *RelativeLayout* con la lista con los diferentes test, contaba con un *Button* para poder acceder a los test.

Para la lista de los test se realizo mediante una nueva *Activity* del tipo *ListView* de modo que pudiese contener tantas preguntas como el test tuviese. Aquí solamente aparecía el nombre de la pregunta y para poder acceder a sus caracteristicas había que pulsarla. Una vez pulsada la pregunta y dependiendo de que pregunta se tratase se creaba una activity nueva, en la Figura 4.3 se puede ver que es una pregunta del tipo *TrueFalse*, esta interfaz era de

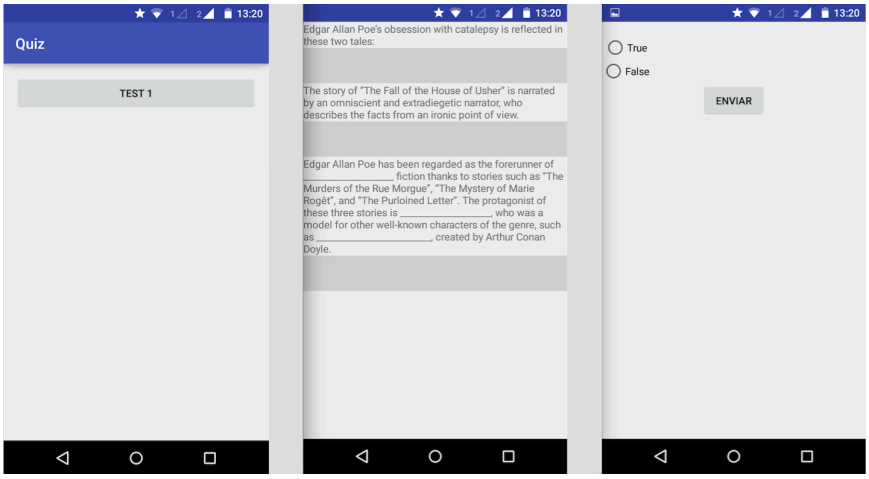


Figura 4.1: Diseño del primer prototipo.

tipo *LinearLayout* que contenia un *TextView* con dos *RadioButtons* para el verdadero o falso y con un boton para enviar.

4.2. Parser

Un parser es un elemento encargado de convertir el texto de entrada en otras estructuras las cuales serán más utiles para el posterior analisis.

En nuestro caso el parser se encarga de recibir el documento, realizado por terceros, con formato .csv estructurado con la información necesaria: Número de pregunta, la pregunta, tipo de pregunta, respuestas y feedback. Es muy importante que todos los documentos .csv tengan la misma estructura, ya que de no ser así puede haber problemas a la hora introducirlos en la base de datos.

En la Figura 4.2 podemos ver el primer test incluido en la aplicación, en la cual podemos observar su estructura en función de su parte mas importante, que es el tipo de pregunta, ya que esto nos dirá como el programa tiene que leer cada linea.

La lectura del fichero no era complicada. Se realizó una clase llamada `ParserCsv` con un método estático llamado `LeerFicheroCsv` que recibía el flujo de entrada y devolvía una estructura de datos del tipo *ArrayList* con objetos de tipo `Question` (serán explicado a continuación). La implementación del método consitía en ir leyendo cada linea del fichero .csv. Dentro

nº	PREGUNTA	TIPO DE PREGUNTA	RESPUESTA	FEEDBACK	X
1.-	Edgar Allan Poe's obsession with catalepsy is reflected in the	Multiple Choice	a) "The Murders of the Rue Morgue" an	a) These tales are two of the detective storie	X
			b) "Berenice" and "Ligeia"	b) Although "Berenice" does revolve around	X
			c) "The Premature Burial" and "Berenice	c) CORRECT. Both "The Premature Burial" ar	X
2.-	The story of "The Fall of the House of Usher" is narrated by a	True / False	FALSE. "The Fall of the House of Usher"	is described by a first person narrator, who h	X
3.-	Edgar Allan Poe has been regarded as the forerunner of ____	Filling the gaps	DETECTIVE		X
			AUGUST DUPIN		
			SHERLOCK HOLMES		

Figura 4.2: Primer test introducido en la aplicacion en formato .csv.

de este cada celda estaba separada por un ';'por lo que esto nos sirve para identificar cada celda hasta llegar al final de linea. La primera celda consistia al numero de pregunta, la segunda al titulo, la tercera al tipo de pregunta y una vez reconocido esto, identificamos el tipo. En nuestro caso en esta primera versión teniamos tres tipos de preguntas

- *Multiple Choice*: Ante este tipo de pregunta hay varias respuestas diferentes y en nuestro fichero está representado por varias lineas que pueden ser correctas o incorrecta, que leemos hasta que encontramos una linea con un número nuevo de pregunta.
- *True/False*: En este caso solo será una linea ya que la pregunta será correcta o incorrecta.
- *Filling the Gaps*: Dependiendo del número de huecos a rellenar habrá que leer una o más líneas hasta llegar a una línea nueva con número de pregunta.

La última celda corresponde al *Feedback* que es un complemento o comentario aportado a cada respuesta. Por lo tanto habrá una linea de *Feedback* por cada respuesta.

Una vez recogidos todo los datos de una pregunta creamos un objeto de tipo `Question` y lo incluimos en la estructura de datos (*ArrayList*)

Todo este proceso se repite por cada pregunta incluida en el documento .csv

Hecho todo esto a continuación se explica la estructura de clases utilizada para administrar todas las preguntas.

4.3. Test

La manera en la que los test están organizados se puede ver en la figura 4.3 en la que observamos una clase padres *Question*, de la cual heredan tres clases en funcion de los diferentes tipos de preguntas *Multiple Choice, TrueFalse y FillingGaps*.

La clase padre *Question* tiene los atributos número de pregunta, nombre y tipo. También tiene los métodos (gets y sets) para acceder a estos atributos.

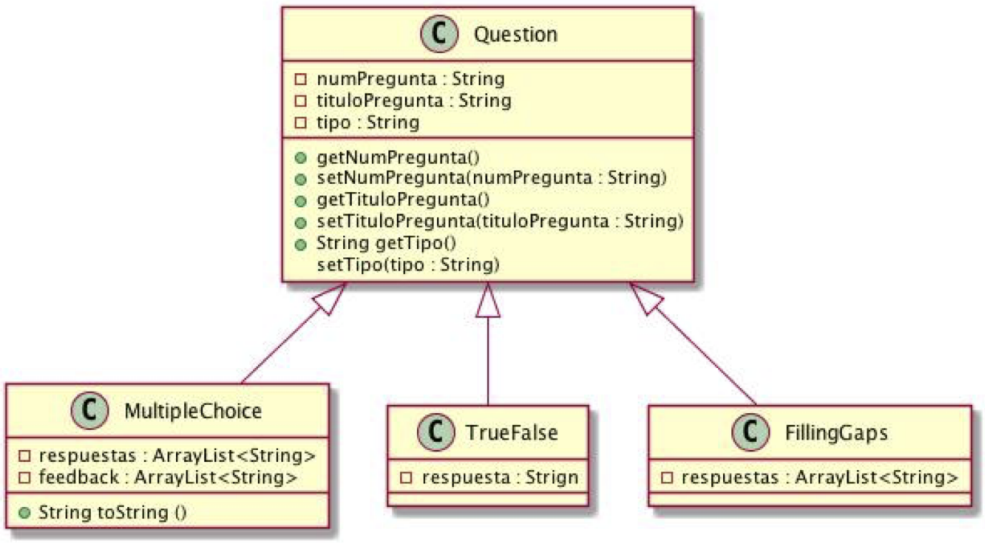


Figura 4.3: Diagrama de clases ultilizado para cargar las preguntas.

En cuanto a las clases hijas, complementan al padre y se diferencian unas de otras en la o las respuestas que pueden tener. En la clase *MultipleChoice* vamos a necesitar una estructura de datos *ArrayList* para poder almacenar las diferentes respuestas y tambien otro *ArrayList* del *Feedback* que nos da una pequeña descripción de cada pregunta. En el caso de la clase *TrueFalse* solamente necesita un boolean para saber si la afirmacion es correcta o no y por último en el caso del tipo *FillingGaps* solamente necesitamos una estructura de datos *ArrayList* para las respuestas ya que no hay *Feedback* en este tipo de pregunta.

De esta manera, y como he explicado anteriormente, cada vez que el parser lee un tipo se crea un objeto de dicho tipo con sus carecterísticas.

Esta arquitectura será modificada en futuras versiones, ya que esta no es del todo funcional, aunque cumple unos requisitos mínimos.

4.4. Dificultades encontradas

No se han entrado demasiadas dificultades a la hora de realizar este primer prototipo más que, aprender a utilizar la tecnología que va a ultilizar durante la realización del proyecto, el etorno Android Studio y el uso de la API de google para aplicaciones Android.

Hay que destacar que se encontraron problemas con la codificación de los caracteres en los ficheros excel y marcarlo como un posible problema futuro a la hora de añadir los test.

Capítulo 5

Primera versión

Atreveos: el progreso solamente se logra así.

Victor Hugo

Resumen: En esta primera versión ya funcional y mucho mas avanzada que los cambios son significativamente grandes, aunque lejos de la teorica versión final.

Como cambios más significativos, se ha incluido la gestion de usuarios mediante bases de datos remotas. También se ha diseñado la base de datos para los test. Otro de los requisitos que no estaba incluido en el primer prototipo es la posibilidad de estudiar a los autores directamente en la aplicación. Cabe destacar tambien la implementación de una plataforma web, a traves de la cual los alumnos pueden incluir directamente los test en la aplicación móvil. Anteriormente los test eran añadidos desde la aplicación como ficheros csv. pero en esta versión la plataforma web se encarga de recibirlos como ficheros de word (.doc o .docx) con una estructura definida, dejandolos a disposición de la Base de Datos.

Tambien se comentaran los problemas que se han ido surgiendo a lo largo de esta versión.

5.1. Nuevas funcionalidades

Podemos hablar a partir de esta versión como una plataforma para la enseñaza de literatura inglesa ya que se puede diferenciar en dos partes y no únicamente de la aplicación android de la que hemos hablado en el capítulo anterior.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra
1	nombreUsuario	varchar(255)	latin1_swedish_ci		No	*Ninguna*	
2	password	varchar(255)	latin1_swedish_ci		No	*Ninguna*	

Figura 5.1: Tabla para los usuarios en la base de datos.

- La primera y más importante es la posibilidad de poder incluir los test directamente por los alumnos de grado y master de estudios ingleses de UAM, en la aplicación. Para esto se ha implementado una página web sencilla, que permite subir un documento con una estructura previemente definida, mostrar al alumno toda la información, que verifique si es correcta y en caso de serlo incluirla en la base de datos

- La segunda concierne a la aplicación y es la de facilitar el estudio de diversos autores o épocas, relacionados con la literatura inglesa y la posibilidad de realizar los test de estos.

Todo esto mediante un sistema de gestión usuarios, para que también en futuras versiones tantos estos como los profesores en caso de desearlo puedan obtener los resultados y puedan ver su progreso y estadísticas.

5.2. Base de datos

Para la realización de la base de datos, después de pasar por diferentes opciones que serán explicadas en el siguiente apartado, hemos acabado usando un sistema gestor de bases de datos JDBC, a través del cual podemos crear un cliente que puede conectar con la base de datos, ejecutar instrucciones SQL y procesar el resultado de estas instrucciones directamente desde el código en Java, sin la necesidad de código en el servidor. El servidor MySQL se encontraba alojado en un el host *sql2.freemysqlhosting.net* con dos tablas una para los usuarios y otra para los test.

El diseño de la base de datos para los usuarios es sencilla como se pude ver en la Figura 5.1. Los usuarios solo constaban de nombre de usuario `nombreUsuario` y constraseña `password`. Siendo ambos no nulos y el nombre de usuario como clave primaria.

Para los tests se han tenido en cuenta los autores y los tres tipos de preguntas:

- *True/False*: Tiene un título, una respuesta que incluye si es verdadera o falsa y un comentario explicando esta pregunta

- *Multiple Choice*: Tiene un titulo, varias respuestas, de cada respuesta se indica si es verdadera o falsa y un comentario por cada una.
- *Fill in the gaps*: Tiene un titulo con huecos libres y por cada hueco libre una respuesta asociada.

En cuanto al diseño de la base de datos para los cuestionarios es más complejo, tal y como puede verse en la Figura 5.2. El Diagrama Entidad Relación consta de dos entidades generales que son `Autores` que tienen `Nombre` (Como clave primaria) y `Descripción`. La relación está definida como un autor puede tener N preguntas así como una pregunta solo puede pertenecer a un autor. La entidad `Preguntas` está definida por los atributos `Id` (Clave primaria de la entidad), `NombreAutor` (Clave foranea con el atributo de Nombre de la entidad `Autores`), `Titulo` y `Tipo`. Esta entidad está definida como una generalización, en las entidades `TrueFalse` que tiene como atributos `omentario`, y un booleano (`Correct`) que indica si la pregunta es correcta. `FillingGaps` esta entidad tiene como atributo un listado de las respuestas (`Listado`) y por ultimo `MultipleChoice` esta ultima entidad tiene una relación con una entidad `PreguntasMult` de 1 a M (una pregunta `MultipleChoice` puede tener varias respuestas), esta nueva entidad tiene como atributos el `Id` (Clave primaria), `IdMultipleChoice` (Clave foranea con la entidad `MultipleChoice`), `Descripción` y un booleano (`Correct`) que indica cual es la correcta.

Se incluye también en la Figura 5.3 el paso del diagrama E/R (diseño conceptual) al modelo relacional (diseño lógico). Se puede observar que como resultado de la generalización de la tabla `Pregunta` se ha generado una tabla diferente.

Se ha escogido este diseño de la base de datos para los test aunque pueda haber redundancia en algunos tipo de dato, presenta facilidad para la mantenibilidad del código y si en futuras versiones se desease añadir algún tipo de pregunta simplemente habría que añadir una nueva entidad a la generalización de preguntas y crear su tabla con los diferentes a atributos

5.3. Plataforma Web

Para la realización de la página web, se ha hecho uso de, principalmente PHP y SQL poder volcar la información en la Base de Datos, aunque también HTML y CSS para el diseño.

Al contrario que en el primer prototipo en el que la información se volcaba a traves de ficheros con formatos .csv, ahora se ha modificado y se volcará a traves de documentos *Word* con formato .doc y .docx

En la Figura 5.4 se puede ver la pagina de inicio (`index.php`) en la que se explican las instrucciones para poder añadir los diferentes test y se incluye un enlace a traves del cual se puede descargar un fichero plantilla. Al final de

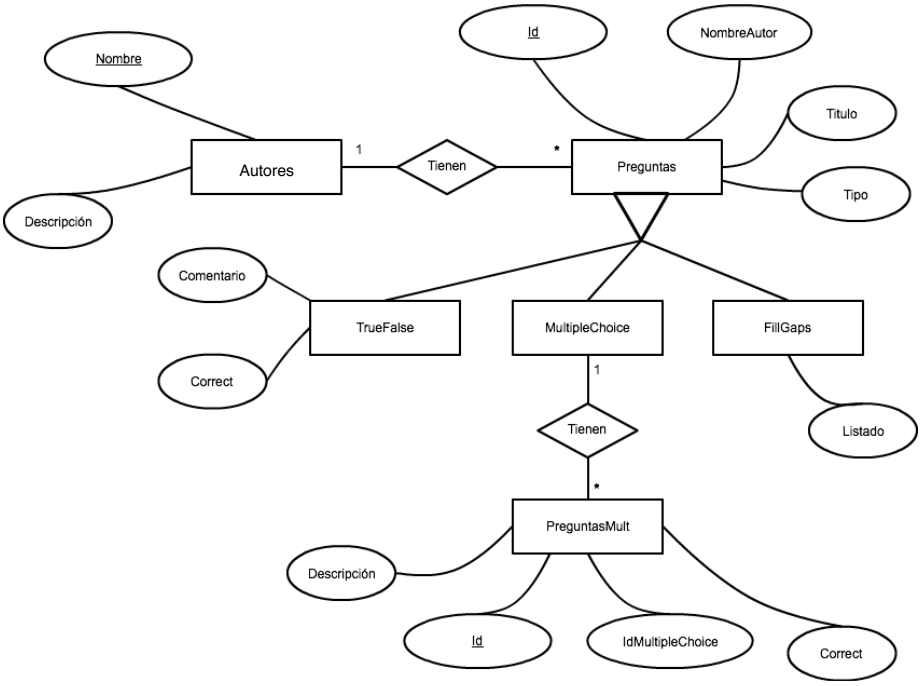


Figura 5.2: Diagrama Entidad Relación de la base de datos de los test.

esta página se incluye un boton del tipo *"multipart/form-data"* que permite al usuario subir un fichero desde su propio ordenador. Al enviar el fichero, si su extensión es de tipo .doc o .docx y su tamaño no supera lo 1024 KB será aceptado y subido al servidor. En este punto el servidor se encargar de coger el fichero y parsearlo, en caso de que el fichero este debidamente estructurado, en una nueva vista permite al usuario verificar la información que se va a ser incluida en la Base de Datos. Si el usuario acepta esta información, esta se añade a la base de datos y se deja a disposición de la aplicación.

5.3.1. Cambios en el parser

Las modificaciones que caben destacar del parser son el cambio de tipo de fichero, ya que antes se realizaba con un fichero .csv y esta vez recibimos un documento de word (ya sea .doc o .docx) y en esta versión la aplicación no será la encargada de parsear el documento, se realiza en lado el servidor (mediante de lenguaje PHP), abrirá el documento word, e irá leyendo línea por línea hasta llegar al final. Cabe destacar también el cambio del lenguaje, ya que antes se creaba un objeto por cada pregunta. Este caso cada vez que se reconoce un tipo diferente se realiza una cadena de texto con el insert en la tabla adecuada.

Cuando el usuario acepta que toda la información es correcta se realiza

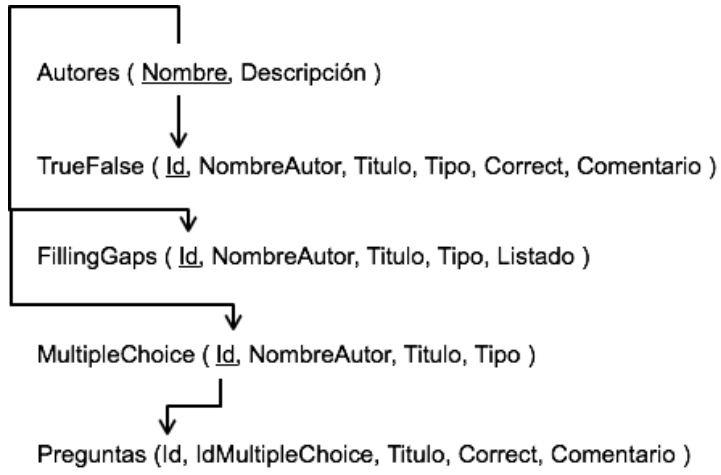


Figura 5.3: Diseño lógico de la base de datos de los test.

un query multiple (*multiquery*) de todos lo insert que se han ido acumulando.

Se ha incluido un elemento verificador que en caso de detectar algún tipo de dato incorrecto o anomalía, ya sea porque el fichero no sigue la estructura indicada o debido a una mala lectura, no permitirá su inclusión en la base de datos ya que pone en peligro y compromete todos datos contenidos en ella. Es por eso que el verificador en caso de notar alguna anomalia no permitirá incluir la información.

Si todo va bien se mostrara toda la información debidamente estructurada para que el usuario pueda verificarla, si este acepta que es correcta realiza una *query multiple* (*multi query*) de todos lo insert que se han ido acumulando.

De esta manera dejamos volcada toda la información en la base de datos y disponible a la aplicación

5.4. Dificultades encontradas

5.4.1. Con la base de datos

Como se ha mencionado anteriormente se encontraron diversas dificultades, posiblemente debido al desconocimiento de la plataforma, ya que se comenzó a realizar la base de datos que Android pone a disposición de sus programadores, *SQLite*, pero esta fue descartada ya que no ofrecía acceso remoto que era el principal uso que queriamos darle para la gestion de usuarios y test.

Figura 5.4: Página Web para la introducción de test.

Finalmente se optó por utilizar el sistema gestor de bases de datos *mySQL*, a pesar de hubo algunos problemas con las librerias en el entorno AndroidStudio que acabaron siendo solucionados.

5.4.2. Con el cambio de documentos

Un cambio en los test enviados por los alumnos, hizo que nos tuviésemos que plantear unos cambios serios que hicieron que la estructura de la aplicación y los requisitos cambiasen. Se espereba recibir unos documentos .csv por pero debido a sus necesidades realizaron en documentos de word.

Una de las diferentes soluciones que intentamos llevar a cabo fue el uso de las librerías POI que Apache pone a nuestra disposición pero mientras con el IDE Eclipse sin problemas, Android Studio no nos lo permitía ya que estas librerías hacían muy pesado el proyecto (superaban las 65.000 lineas de código).

Despues de pensar en diferentes soluciones se optó por la creación de la plataforma Web y realizar el parser con PHP que trabaja muy bien con documentos de word. De esta forma también dejamos más libre la aplicación, que únicamente tiene que hacer las consultas a la Base de Datos.

Capítulo 6

Segunda versión

La vida es una serie de colisiones con el futuro; no es una suma de lo que hemos sido, sino de lo que anhelamos ser.
José Ortega y Gasset.

Los cambios que se han realizado en esta nueva versión son numerosos, sobre todo en diseño y en la forma de realizar los test. Aunque queda funcionalidad por implementar, esta nueva versión puede considerarse clave ya que fue enviada a los alunmos para que evaluasen su funcionamiento y realizasen un formulario o encuesta para poder evaluar los resultados y opiniones de una manera estadística, para encontrar fallos y sugerencias para futuras versiones.

Respecto a los cambios en el diseño, se ha utilizado *Material Design* y recursos de este para hacer un diseño mas agradable e intuitivo para el usuario.

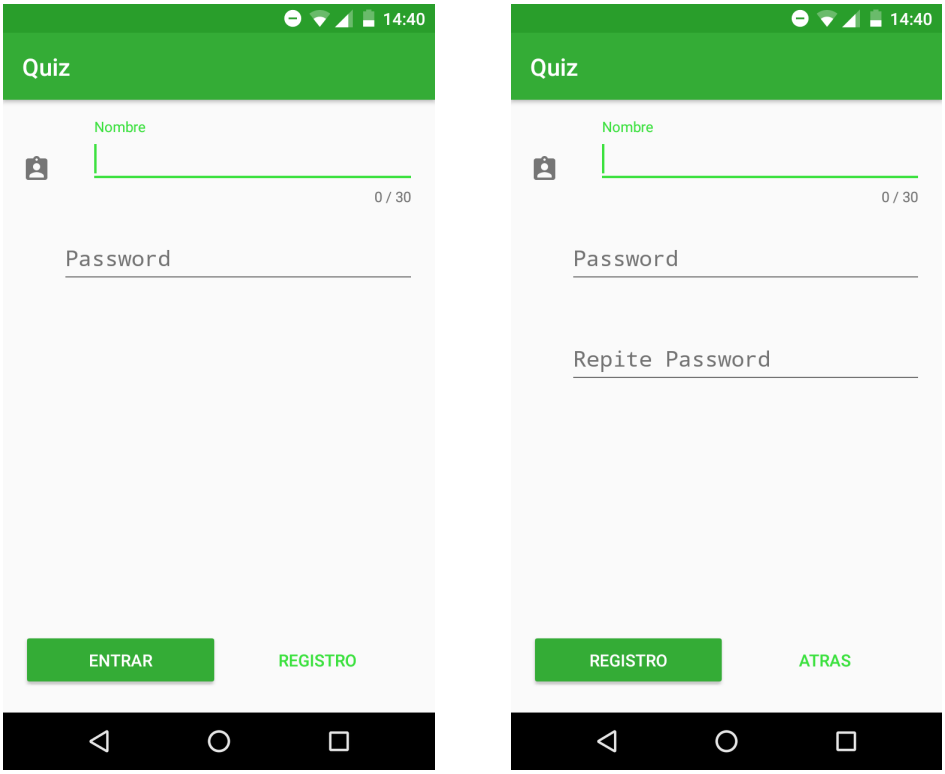
El diseño de la base de datos no ha sido modificado. Las únicas modificaciones en este aspecto han sido respecto al acceso a los datos, ya que a partir de ahora se realizarán a traves de un servicio Web con PHP.

La realización de los test ha sido modificada dejando atras la lista de preguntas e implementando, a traves de fragmentos, un sistema para realizar cada pregunta de forma independiente.

Se hablará tambien de los problemas encontrados a lo largo de esta versión así como solución y propuestas para nuevas versiones.

6.1. Cambios en el diseño

Como hemos dicho, en esta versión uno de los cambios mas significativos ha sido en diseño, implantándose elementos de *Material Design*, ya sea para mejorar el aspecto visual, colores, forma de presentar la información y ani-



(a) Login para usuarios ya registrados (b) Registro para nuevos usuarios

Figura 6.1: Capturas de pantalla del Login y Registro de la aplicación.

maciones como tambien elementos para mejorar la eficiencia de la aplicación.

Material Design es una guía exhaustiva para el diseño visual, como tipografía, *grids*, espacios y colores poniendo mucho enfasis no solo en el diseño estático de los elementos de la interfaz de usuario, si no también en el movimiento de éstos por la pantalla, intentando reforzar la experiencia del usuario.

Es por eso que *Google* ha creado la librería *Android Design Support* incluye los elementos para la buena implantación de los principios de *Material Design*, facilitando en gran medida su implementación, permitiendonos así centrarnos más en la funcionalidad principal nuestra aplicación, y no tanto en su comportamiento puramente visual. *Google* también se ha encargado de añadir soporte para versiones anteriores a Android 5.0.

En la Figura 6.1 podemos ver los *layouts Login* (a) y el *Resgistro* (b). Ambos *layouts* están compuestos de *CoordinatorLayout*, este tipo de *layout* perteneciente a la libreria *Android Design Support* de la que hemos hablado anteriormente, va a actuar como contenedor principal y es el encargado de la animación de algunos de los elementos de la interfaz. Podemos observar

esto cada vez que seleccionamos uno de los elementos *EditText* tienen estos *layouts* mostrando una pequeña animación.

Para mostrar las listas de autores y de test (Como podemos ver en la Figura 6.2), ambos están contenidos dentro de un mismo *layout*, pero el *widget* (elemento) más importante es el *TabLayout*. Este componente, contenido en la libreria de diseño *Android Design Support* que proporciona *Google* es el encargado de facilitar el cambio entere un contenido u otro mediante pestañas y el desplazamiento entre ellas dentro de un mismo actividad.

Como vamos a trabajar con elementos (listas) diferentes dentro de un mismo *Activity* vamos a introducir un nuevo elemento que nos facilita Android, los fragmentos. Estos son una sección modular de interfaz de usuario contenida dentro de una actividad anfitriona permitiendo versatilidad y optimización de diseño. Se trata de miniactividades contenidas dentro de una actividad anfitriona, manejando su propio diseño (un recurso layout propio) y ciclo de vida.

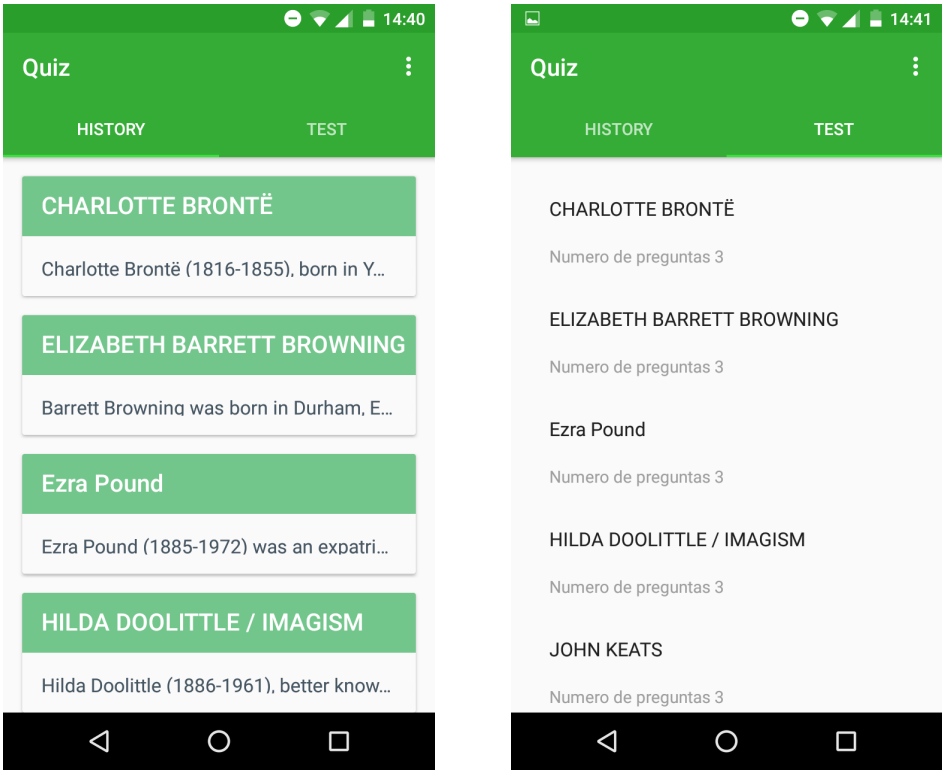
Cada lista, un fragmento diferente, utiliza el *Widget RecyclerView* (una versión más flexible y avanzada de *ListView*) añadido en la librería de *Material Design*. Este *Widget* supone un elemento fundamental, ya que permite por un lado añadir *Cards* (Tarjetas) personalizadas con facilidad y por otro mejorar la eficiencia de la aplicación para soportar grandes listas de elementos. Es decir, cada *RecyclerView* tiene tantos *CardView* como elementos tenga la lista.

Para nuestra aplicación hemos realizado dos *CardView* diferentes

- *CardView* de los autores: Compuesto un *ImageView* (aunque en esta versión no hay imagenes esta puesto para poder soportarlas en futuras versiones), y por dos *TextView*, uno para el título y otro para mostrar un poco de texto.
- *CardView* de los test: Solo esta compuesto por dos *TextView*; uno para el título y otro para el numero de preguntas de cada test.

Para mostrar la información de cada autor (Figura 6.3) siguiendo las guias de diseño y elementos que nos proponen en *Material Design* dentro de la libreria *Android Design Support* cabe destacar dos *Widgets*, por un lado *AppBarLayout*, que contiene un *CollapsingToolbarLayout* que permite la animación y por otro lado el *Widget NestedScrollView* que muestra toda la descripción del autor y que permite el desplazamiento en caso de ser lo suficiente largo.

Para la realización de las preguntas se optó por una *fragment* diferente dentro de una misma actividad, y al responder cada pregunta en caso de que esta estuviese contestada erroneamente se utilizada un *Dialog*, como podemos ver en las Figuras 6.4 (b) para mostrar un comentario sobre la solución correcta y permitir al usuario aprender. En caso de estar contestada



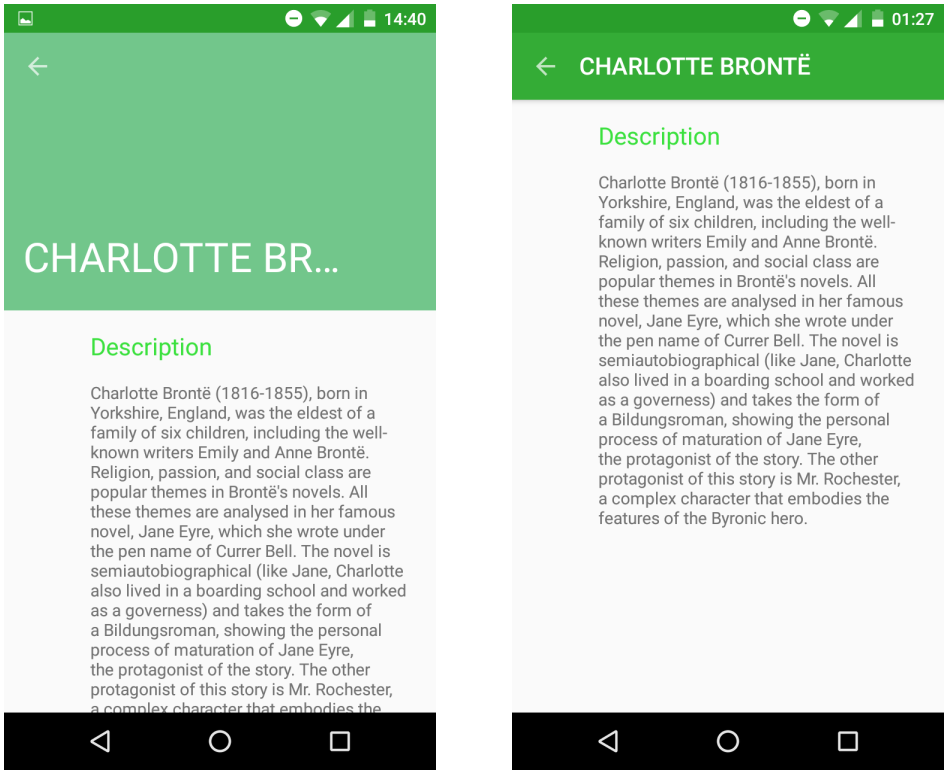
(a) Lista con los diferentes autores (b) Lista con los diferentes Test

Figura 6.2: Main del aplicación.

correctamente se muestra un *Toast*, como se puede ver en la Figura 6.5 comunicando al usuario que su respuesta es correcta y se pasa a la siguiente pregunta.

En las preguntas de *True/False* se utilizó un *RelativeLayout* y, como elemento mas importante un *RadioGroup* con dos *RadioButton* para las opciones de *True/False*

En las de *Filling The Gaps*, como podemos ver en la Figura 6.6 (a), la estructura del *Layout* era más compleja ya que cabe destacar el uso de un *ListView* que se rellena de modo programático ya que los huecos a rellenar dependen de cada pregunta, siendo una línea para una parte del enunciado y la segunda línea un hueco a rellenar, y cuando el usuario hace click sobre un espacio a rellenar se abre un *Dialog List* que muestra las opciones disponibles, como se puede ver en la Figura 6.6 (b), por lo que cuando se seleccionaba una opción se rellena en la lista de la pregunta. Si todos los huecos no estan rellenos y el usuario intenta pasar a la siguiente pregunta, la aplicación muestra un *Toast* comunicando que faltan huecos por rellenar. Cuando están todos lo huecos rellenos se sigue el mismo procedimiento que



(a) Lista con los diferentes autores (b) Lista con los diferentes Test

Figura 6.3: Layouts de la informacion de cada autor.

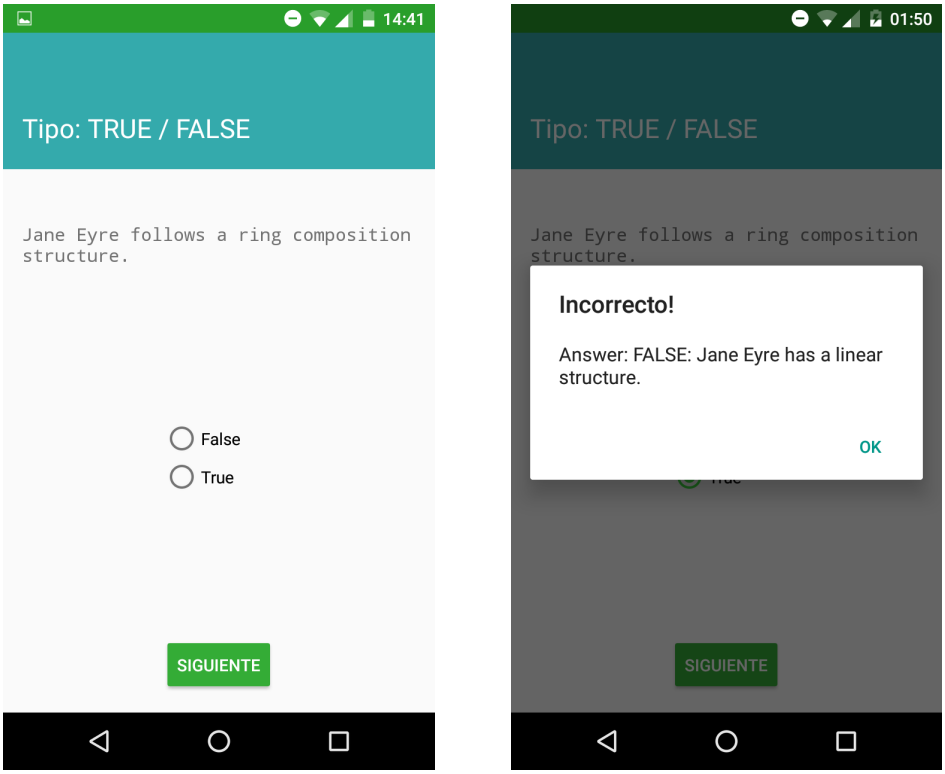
en la pregunta anterior. Si es correcto, muestra un *Toast* (ver Figura 6.5) y pasa a la siguente; si no, muestra un *Dialog* (ver Figura 6.11).

Por último las preguntas de *Multiple Choice*, Figura 6.7 (a) son tres *CheckBox* independientes que tienen que estar seleccionado al menos uno para poder pasar a la siguiente pregunta, realizando el mismo procedimiento que en las anteriores (ver Figura 6.7 (b))

6.2. Cambios en la base de datos

En cuanto al diseño de la base de datos este no ha sido modificado. Los únicos cambios que hay que mencionar son en el acceso a los datos pasando del SGBD (Sistema Gestor de Bases de Datos) a un servicio Web.

Este servicio Web accede a los datos de un servidor externo a traves de Php, MySql y Json. Para ello se ha utilizado la libreria *Volley* de Google para otimizar el envío de peticiones Http desde aplicaciones Android hacia servidores externos. Este componente actúa como una interfaz de alto nivel, liberando al programador de la administración de hilos y procesos tediosos



(a) Layout pregunta True False

(b) Layout pregunta True False con Dialog

Figura 6.4: Layouts de la pregunta True False.

de parsing, para permitir publicar fácilmente resultados en el hilo principal. Al estar basado en peticiones, se ha implentado el patrón de *Singleton*. Con esto, al limitar el alcance de la clase a un solo objeto, la cola de peticiones será la misma en todo el proyecto.

Esta migración solo se llevó a cabo con los usuarios, dejándose el acceso a los test con el SGBD, para realizarlo en futuras versiones.

6.2.1. Código PHP

Para la conexión con la base de datos Mysql se ha utlizado una conexión PDO, que nos permite proteger los datos de inyecciones SQL.

Y mediante scripts PHP se gestionan las peticiones, para posteriormente parsear los datos en formato Json y que nuestra aplicación Android interprete estos resultados de forma legible.

Para ello se ha creado una clase (`Database.php`), con un patrón *Singleton* para limitar el número de aperturas a la base de datos a una sola. Esta tiene todos los métodos encargados para conectarse a la base de datos y necesita

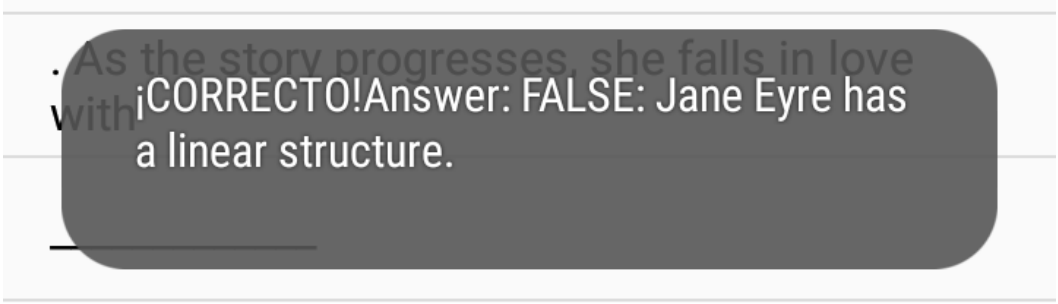


Figura 6.5: Ejemplo de *Toast* cuando el usuario responde correctamente una pregunta.

de otra `mysql_login.php` que provee los datos necesarios para conectacte a la base de datos como nombre del Host, nombre de la base de datos, nombre del usuario y contraseña.

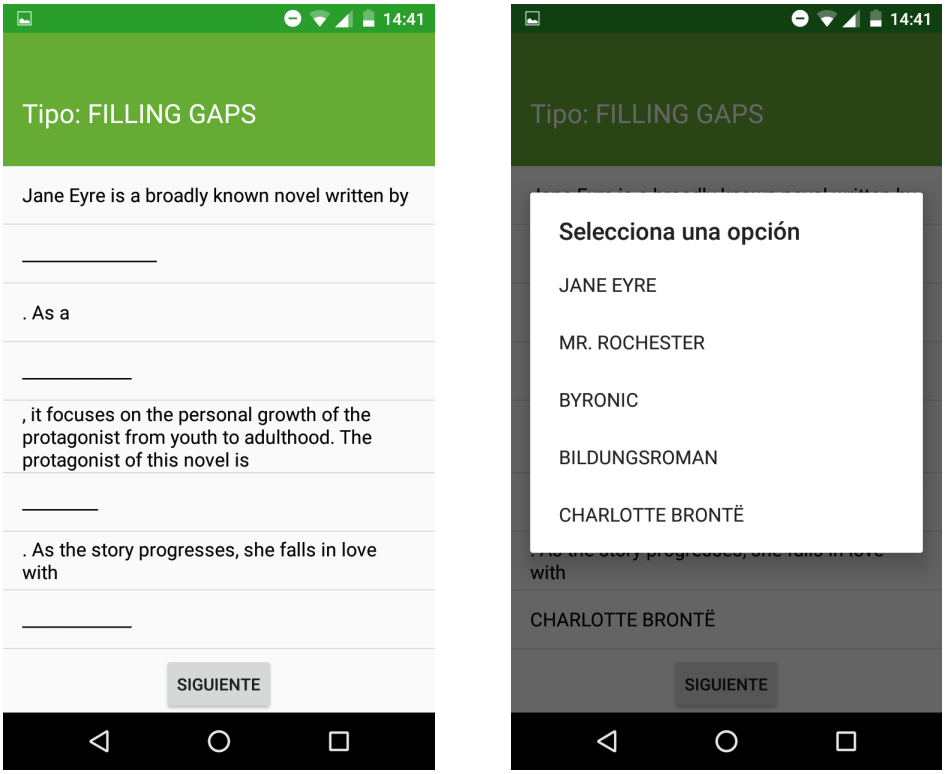
Para acceder a los datos se han creado dos clases,

- `check_user.php`: Se encarga de realizar una consulta (*Select*) en la base datos, devolviento un objeto Json, con información sobre la consulta (si ha habido un problema a la hora de realizar la conexión con la base de datos o si el usuario existe) a la aplicación que posee un método `procesarRespuesta(JSONObject response)` dentro de la clase `Login.java` que recibe el objeto Json lo parsea y en caso de que devuelva que el usuario es correcto lanza la actividad principal `MainActivity.java`. En caso contario muestra un *Toast* indicando que el usuario no existe o que la contraseña introducida es erronea.

- `insertar_usuario.php`: Tiene un comportamiento similar al anteriormente explicado, aunque este se encargar de registrar usuarios realizando una consulta de inserción (*Insert*). Si la consulta da error al introducir el nombre de usuario, quiere decir que ya existe ese nombre usuario en la base de datos (el nombre de usuario es clave primaria en la tabla de los usuarios). Si devuelve exito, el usuario ha sido creado con exito. Se devuelve un objeto Json a la aplicación que con el método `procesarRespuesta(JSONObject response)` de la clase `RegistroActivity.java` parsea el objeto y muestra un *Toast* con el resultado de la insercción

6.3. Cambios en el código de la aplicación

En esta sección se explica como se han implementado los cambios realizados en el diseño.



(a) Layout pregunta Filling the Gaps

(b) Layout pregunta Filling the Gaps con ListDialog

Figura 6.6: Layouts de la pregunta Filling the Gaps.

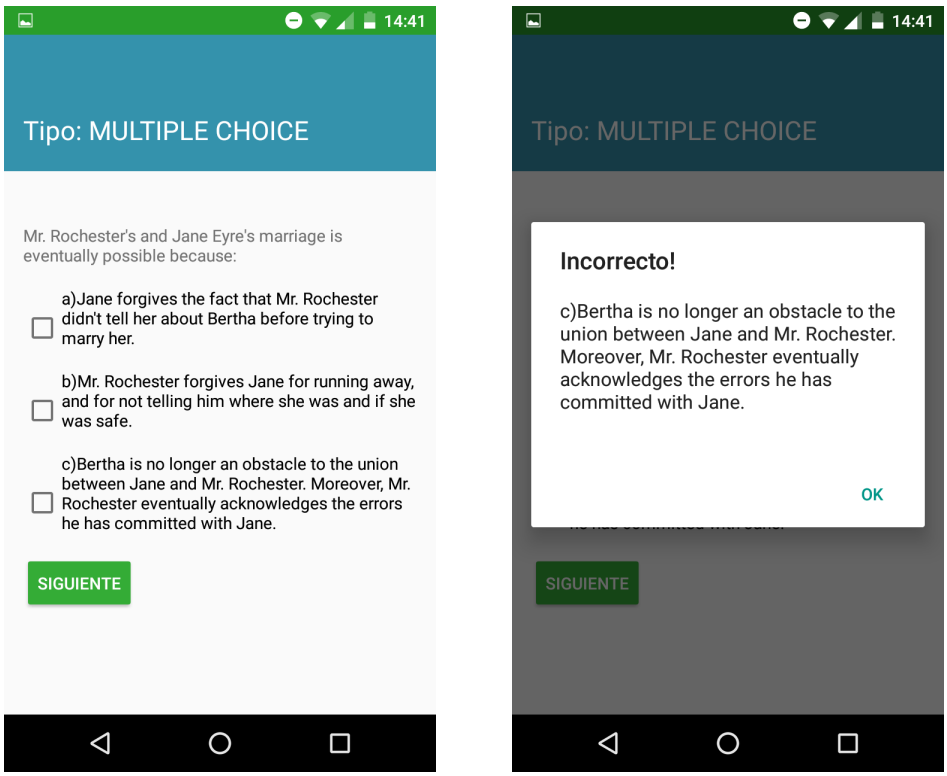
6.3.1. MainActivity

En la Figura 6.8 se puede ver el diagrama de secuencia que muestra la creación de la *Activity* principal la cual comienza con la llamada al método `onCreate()`

A traves de la creación de un objeto `Toolbar`, perteneciente a la librería *TabLayout* de *Material Design*, conseguimos dividir el contenido en subcategorías.

Con el método *setupViewPager* creamos un *Adapter* que va a contener las diferentes subcategorías y cada una de estas es un fragmento diferente. Este *Adapter* extiende de la clase *FragmentPagerAdapter* que proporciona los métodos necesarios para la gestión de cada subcategoría.

Para realizar un fragmento es necesario crear una clase que herede de la clase *fragment* que Android nos facilita. En nuestro caso vamos a crear dos, uno para la parte de la lista de la historia, `HistorySectionFragment.java` y un segundo para la lista de los test `TestSectionFragment.java`. La diferencia entre los dos se puede ver en la Figura 6.2, para las historias se muestra



(a) Layout pregunta Multiple Choice

(b) Layout pregunta Multiple Choice con Dialog

Figura 6.7: Layouts de la pregunta Multiple Choice.

el nombre del autor y un pequeña descrición por cada elemento o *Card* de la lista y para los test el nombre y el número de preguntas que tiene cada autor.

Para cada fragmento se ha implementado el elemento *RecyclerView*, que podemos decir que es la parte más importante y encargada de crear la listas. El *RecyclerView* permite multiples personalizaciones de los elementos de la lista es por eso que hace uso de una clase interna adaptador `ContentAdapter` que sigue el patrón de diseño *view holder*, es decir define una clase interna que extiende de *RecyclerView.ViewHolder* que se encarga conectar con el layout de las *Card* y de los datos de estos, en la Figura 6.9 podemos ver dos ejemplos de las *Cards* para los autores y los test.

El adaptador tiene tres métodos abstractos a los que debemos aplicar el modificador *override*:

- *getItemCount*: Éste método devuelve el número de elementos en la lista.

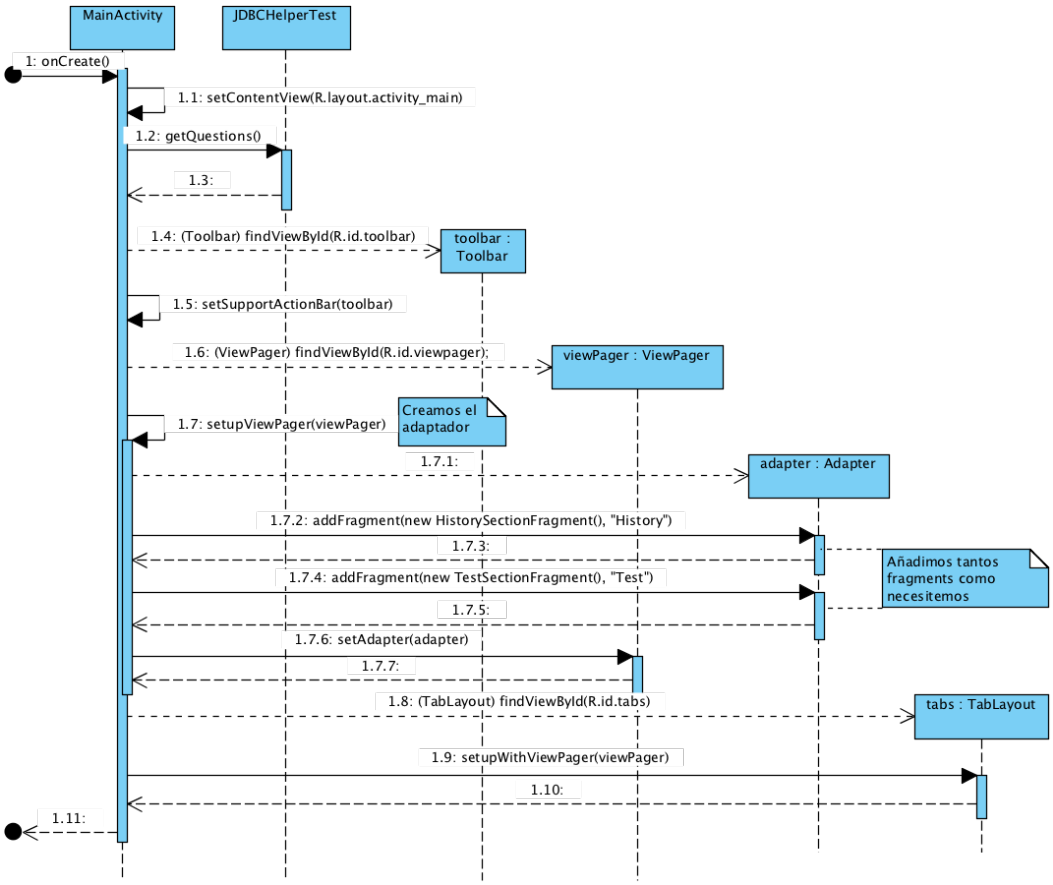


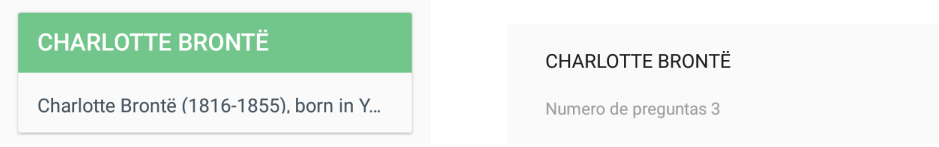
Figura 6.8: Diagramas de secuencia de la clase principal `MainActivity`.

- *onCreateViewHolder*: Éste método es llamado cuando el *ViewHolder* necesita ser inicializado, especificando el *layout* que cada elemento de *RecyclerView* debería usar. Esto se hace al inflar el *layout* y devolviendo el *ViewHolder*
- *onBindViewHolder*: Para especificar el contenido de cada elemento del *RecyclerView*. En nuestro caso tendrá la información del autor que establece los valores del *CardView*

6.3.2. Sistema de preguntas

Podemos considerar la parte del sistema de preguntas una parte fundamental de la aplicación, ya que este es uno de los principales funcionalidades demandados por el cliente.

Para llevarla a cabo se ha creado la clase *TestActivity*. Esta clase, realizada como una *Activity*, será la encargada de gestionar la lógica de las



(a) Ejemplo de *Card* para los autores

(b) Ejemplo de *Card* para los test

Figura 6.9: Main del aplicación.

preguntas, desde mostrarla, corregirla y pasar a la siguiente. Para ello se utilizarán fragmentos y *Dialogs*.

Cuando un usuario ha pulsado sobre un elemento de la lista de los Test, esta crea un *Intent* con la actividad destino, para posteriormente pasarle la posición del elemento seleccionado mediante el método *putExtra()*, que le pasa como primer parametro una *key* que se utiliza en la actividad destino para recuperar la información, que corresponde al segundo parametro. Por ultimo se inicia la actividad con el método *startActivity()*

Cuando inicia o se crea la *Activity* (método *onCreate()*) se le asigna su *layout* vacío ya que será en este donde se irán añadiendo los diferentes fragmentos. Posteriormente se recibe la posición pasada por la actividad anterior y con la llamada al método *getTestByPosition(int position)* se recibe un objeto de la clase *Test* que contiene todas las preguntas de ese autor. Con este objeto se van colsultando una por una todas las preguntas a traves del método *getQuestionByPosition(int position)* de la clase *Test* y se van creando los diferentes fragmentos según el tipo de pregunta.

Para añadir los fragmentos hay que hacerlo mediante programación ya que son dinámicos y no podemos prever cuantos serán. Lo primero que tenemos que hacer es crear una instancia del *FragmentManager* a traves del método *getFragmentManager()*. A continuación creamos una instancia de *FragmentTransaction* para realizar la transacción de fragmentos y con el método *beginTransaction()* indicamos que vamos a realizar una transacción de fragmentos. Por último, se ha utilizado un *switch* para identificar el tipo de pregunta, creamos un objeto del fragmento y lo añadimos a la transacción a traves del método `add()` pasándole el id del *layout* padre que va contener, el fragmento y el objeto del fragmento que mostrara dicha vista. Para terminar la transacción llamamos al método *commit()*. Con la llamada a este método se crea el fragmento y se llama a su método *onCreateView()* que recibe como parámetro el recurso xml padre, al cual será añadido el xml correspondiente al fragmento.

En la Figura 6.10 podemos ver el diagrama de secuencia de esta actividad, en la que se crean los diferentes *fragmentos* en función del tipo de pregunta.

También ha sido necesario el uso de *Dialog* ya que como se ha explicado cada vez que el usuario responde de forma erronea una pregunta la aplicación muestra un *Dialog* indicando que la pregunta es erronea y la solución

correcta. Esto es común en los tres tipos de preguntas. El *Dialog* se ha implementado como una clase que hereda de *DialogFragment*, y hemos sobrescrito el método *onCreateDialog()* que devuelve una instancia del *Dialog*, en nuestro caso del tipo *AlertDialog* (diseñado para mostrar un titulo, un mensaje y hasta 3 botones confirmación) con los atributos *correct* que indica si la respuesta es o no correcta y *comment* para indicar cuál es la correcta y únicamente un botón de confirmación para pasar a la siguiente pregunta. Un ejemplo de *Dialog* lo podemos ver en la Figura 6.11.

6.3.2.1. *TrueFalseFragment*

Como su propio nombre indica, esta clase, que hereda de la clase *Fragment*, será la encargada de las preguntas de tipo *TrueFalse*. Con *inflate* añadimos el *layout* del fragmento en este caso `question_true_false.xml` y con el método *findViewById* obtenemos los elementos del *RadioGroup* para más tarde, cuando el botón de siguiente ha sido pulsado, y con el método `isChecked()`, obtener cuál de los dos (*True* o *False*) ha sido seleccionado y comprobarlo con el objeto de la pregunta *TrueFalse* para corregirlos.

El sistema corrige la pregunta y muestra un *Dialog* en caso de que este mal respondido o un *Toast* en caso contrario.

6.3.2.2. *MultipleChoiceFragment*

Para las preguntas de tipo *MultipleChoice* se ha creado una clase que hereda de fragemento. Se comienza obteniendo el *ViewGroup* con la llamada al método `inflate()` y asignándole el *layout* `question_multiple_choice.xml` y obteniendo los elementos de este con la llamada a `findViewById()`. Con la referencia a los elementos del *layout* y con la pregunta correspondiente se añade la información, titulo del pregunta, posibles respuestas... para mostrarlo al usuario.

Cuando se ha respondido a enviar la respuesta se procede de la misma manera que el resto de tipos; el sistema verifica que se ha seleccionado al menos una respuesta, se comprueba si es correcta y muestra al usuario la corrección a traves de un *Dialog* o un *Toast*.

6.3.2.3. *FillingGapsFragment*

Esta clase, encargada del tipo *FillingGaps*, hereda de la clase *ListFragment*, ya que la forma de presentar este tipo de pregunta es diferente. Cada pregunta de este tipo tendrá diferentes espacios a rellenar que otra, por lo que se decidió que este fragmento heredase de *ListFragment* y el layout *question_filling_gaps.xml* que se añade a la actividad con el método *inflate()* tiene como elemento fundamental un *ListView* que se rellena de forma programática para contener la lista.

En este fragmento hay que destacar varios elementos:

- *ArrayList* de respuestas desordenadas: Se obtiene de la llamada al metodo `getRespuestasAleatorio()` del objeto de la pregunta *fillingGaps* que devuelve un *ArrayList* con todas posibles respuestas desordenadas.
- *Dialog* para las opciones: Tiene un objeto *ListDialog* al que se le pasa el *ArrayList* con todas las preguntas desordenadas obtenido anteriormente, como podemos ver en la Figura 6.12, dando la opción al usuario de escoger una respuesta cuando ha seleccionado un hueco rellenar.
- *ArrayList* de huecos contestados: De tamaño igual qué elementos por rellenar, al que se le irán añadiendo los elementos una vez que el usuario los vaya seleccionando del *ListDialog*.
- *ArrayList* con la solución correcta: Este se obtiene a traves del método `getRespuestasCorrectas()` del objeto de la pregunta *fillingGaps*.

Cada vez que el usuario seleccione un elemento o hueco a rellenar se llama al método `onListItemClick()` que ha sido sobrescrito de la clase padre, pasandole la posición. Aquí se llama al método `show()` del *ListDialog* explicado anteriormente y se le pasa el *ArrayList* de respuestas desordenadas. Cuando el usuario seleciona una respuesta esta se añade al *ListView* original y al *ArrayList* de preguntas contestadas. Si todas han sido rellenadas y se pulsa el boton siguiente, el sistemas comparará el *ArrayList* con las respuestas correctas y con el de contestadas, mostrando si es o no correcta.

6.4. Dificultades encontradas

Esta versión como se ha comentado fue enviada a los alumnos para que la utilizasen y pudiesen evaluar y entre los diferentes problemas encontrados

- No se publico en la tienda de aplicaciones de Android: A la hora de enviar la aplicación a los usuarios, se opto por enviar directamente el fichero (.apk) de la aplicación, ya que no era una versión definitiva
- Problemas con la codificación de caracteres: Antes de enviar la aplicación a los usuarios, se detectaron problemas los codificación en la información de algunos caracteres como la comillas, signos de interrogación o apostrofes. Esto fue solucionado antes de haberla enviado cambiando la codificación directamente en la base de datos.
- Con la instalación de la aplicación: Se nos comunicó que los usuarios encontraron diversos problemas a la hora de instalar la aplicación.

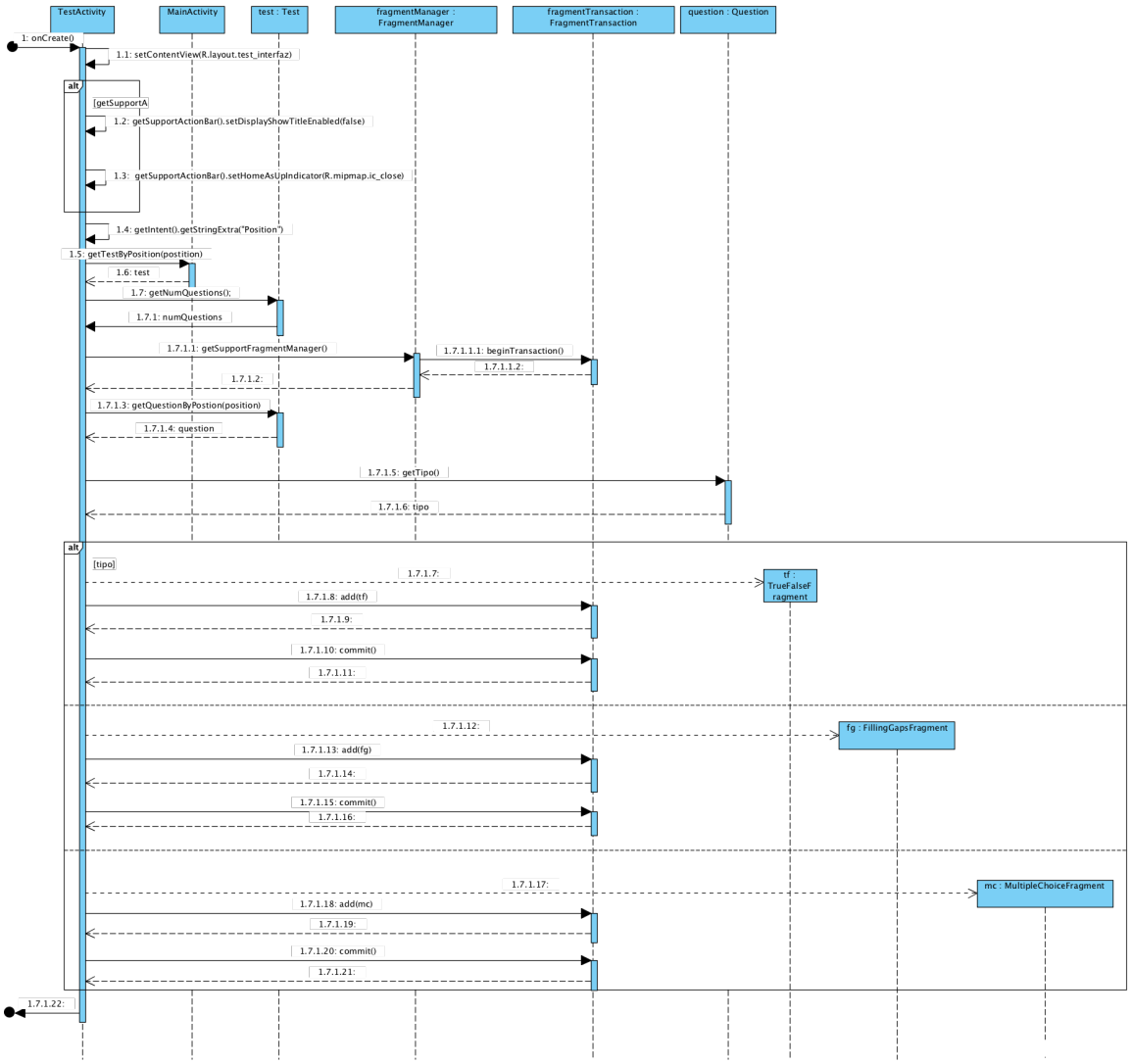


Figura 6.10: Diagrama de secuencia para la clase `TestActivity`

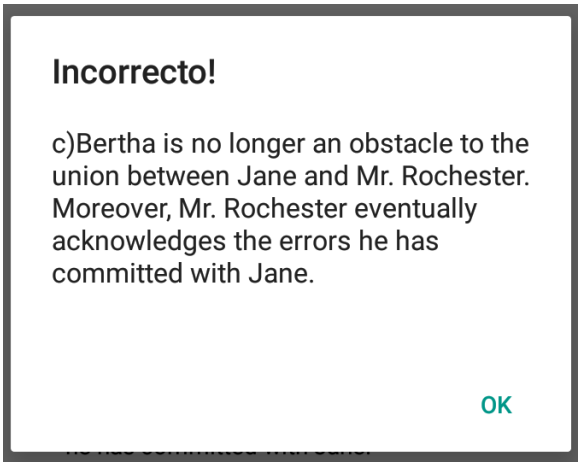


Figura 6.11: Ejemplo de un *Dialog* de tipo *AlertDialog*.

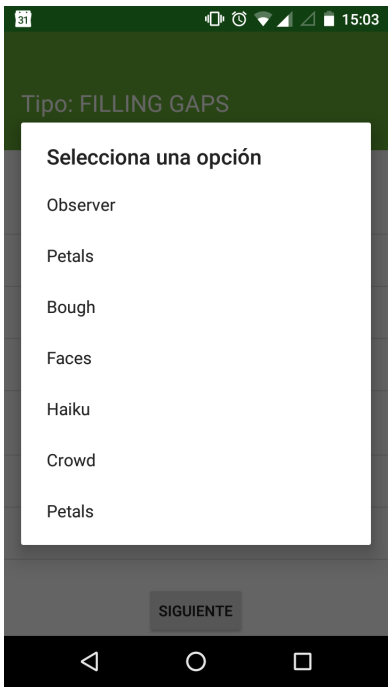


Figura 6.12: *ListDialog* que muestra las diferentes opciones al usuario en el tipo de pregunta *FillingGaps*.

Capítulo 7

Evaluación con los usuarios

La raza humana se encuentra en la mejor situación cuando posee el más alto grado de libertad.
Dante Alighieri

La segunda versión explicada anteriormente, fue una versión que fue enviada a los usuarios, en nuestro caso a la Dra. Ana González-Rivas Fernándezy a los alumnos de grado y master de estudios ingleses de la Universidad Autónoma de Madrid (UAM) para poder obtener una evaluación por parte de estos para que poder encontrar fallos y deficiencias de la aplicación. Por ello se realizarón un encuesta con *Google Forms*. En total se obtuvieron 41 respuestas todas de los alumnos de dichos estudios. Para explicar la encuesta y las preguntas realizadas primero explicaremos los resultados obtenidos de cada pregunta en función del tipo y al final haremos un analisis de los resultados.

Esta encuesta ha sido realizada para evaluar la satisfacción de los usuarios, detectar fallos y problemas para mejorar la aplicación en futuras versiones.

Para obtener un resumen de todas los respuestas, visite el siguiente enlace:

https://drive.google.com/open?id=0By7E_lbRjFmeNThheG5tRTh6OGM

Para obtener los resultados individuales de los usuarios, visite el siguiente enlace:

https://drive.google.com/open?id=0By7E_lbRjFmeR2l3VW1XcnR4T28

7.1. Preguntas de selección múltiple

Para puntuar cada pregunta se ha seguido el criterio de que 1 es malo y 5 como muy bueno

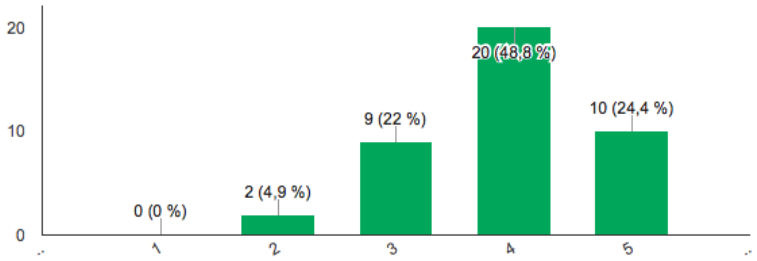


Figura 7.1: La aplicación Android me parece una manera interesante de repasar contenidos vistos en clase.

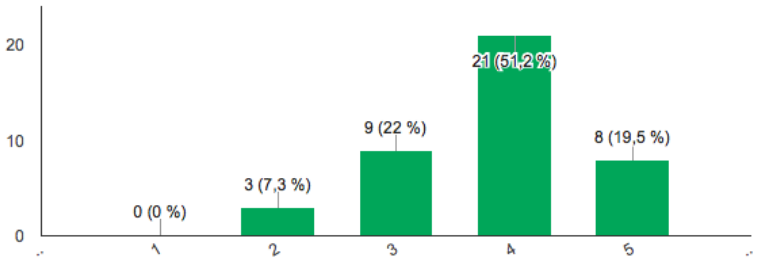


Figura 7.2: La aplicación Android puede constituir para mí una práctica informal para repasar contenidos para un examen.

En la Figura 7.1 tenemos los resultados de la primera pregunta en la que un 50 % (20 usuarios) cree que la aplicación le parece una buena forma de repasar contenidos vistos en clase y casi un 25 % (6 usuarios) la consideran como muy buena. Esta pregunta ha tenido 3.92 de media sobre 5.

En la Figura 7.2 se han obtenido unos resultados similares a la pregunta anterior, 50 % (21 usuarios) bueno y 22 % (9 usuarios) normal para evaluar si la consideran una práctica informal para repasar contenidos. 3.83 de media sobre 5.

En la Figura 7.3, para evaluar si a los usuarios la elaboración de contenido para posteriormente esten en la aplicación disponibles a todo el mundo, les parece una actividad interesante, para ayudarles a estudiar. Un 50 % (20 usuarios) piensan que es una buena idea y un 20 % (8 usuarios) les parece normal. 3.58 de media sobre 5.

Para la evaluación de la facilidad de uso (Figura 7.4), un 40 % (17 usuarios) creen que es facil de usar y un 30 % (12 usuarios) ni facil ni dificil. 3.85

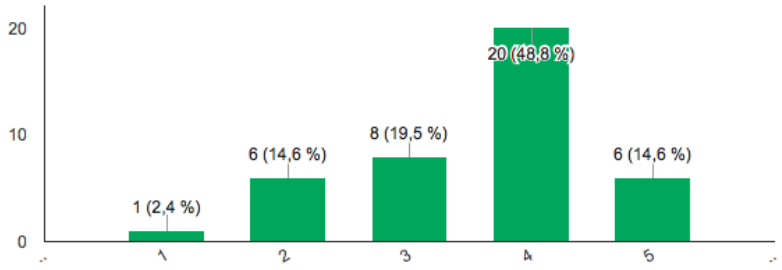


Figura 7.3: La elaboración de contenidos para ampliar la aplicación Android me parece una actividad interesante que puede ayudarme en mi proceso de aprendizaje.

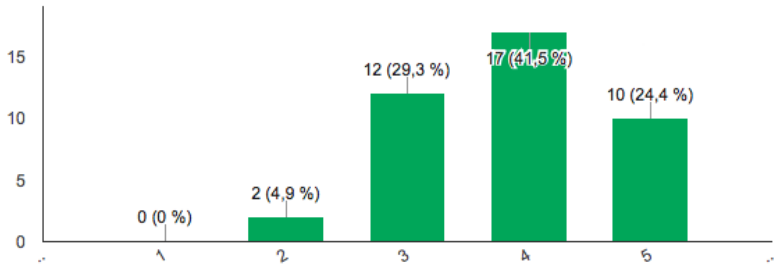


Figura 7.4: La aplicación es fácil de usar.

de media sobre 5.

Respecto a si la interfaz de usuario es agradable e intuitiva, Figura 7.5, un 40 % (18 usuarios) piensan que es normal y otro 40 % (17 usuarios) creen que es buena. 3.41 de media sobre 5.

En la Figura 7.6 se preguntó para ver la satifacción general con la aplicación, un 50 % (21 usuario) opinan que es buena y un 35 % (14 usuarios) les parece normal. 3.48 de media sobre 5.

7.2. Preguntas desplegable

Aquí se le pedía al usuario evaluar si tuvo problemas a la hora de instalar la aplicación, (Figura 7.7). Un 54 % respondio que alguno, un 34 % dijo que ninguno, y un 12 % que muchos

A continuación se les preguntaba por la edad, siendo la gran mayoría

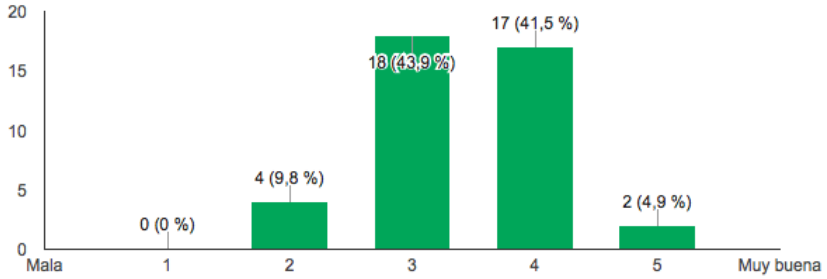


Figura 7.5: Tiene una interfaz de usuario agradable e intuitiva.

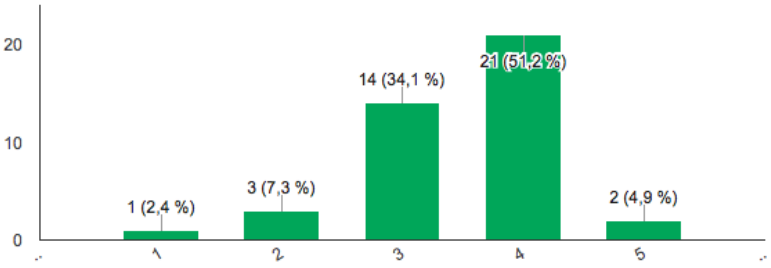


Figura 7.6: ¿Cuál es su nivel de satisfacción general con la aplicación?.

(mas de 75 %) de entre 19 y 21 años, como se puede ver en la figura 7.8.

Se les preguntó por el sexo y como se puede ver en la figura 7.9 casi un 75 % era de sexo feminino.

Por último para este tipo de preguntas se quería saber si todos los que respondian eran alumnos de la Universidad Autónoma de Madrid figura 7.10

7.3. Respuesta larga

Por no meter todas y cada una de las respuestas de cada pregunta se van a describir las respuestas fundamentales y que consideramos más importantes para nuestro proyecto, ya que si se necesita hay un enlace al principio del capitulo para poder verlas.

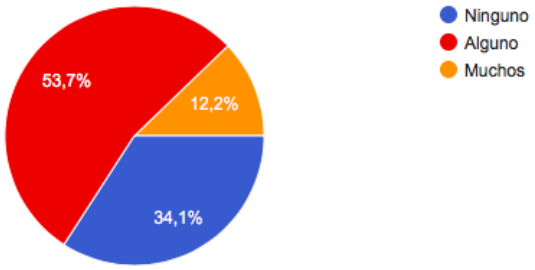


Figura 7.7: ¿Encontraste muchos problemas al utilizar la aplicación?.

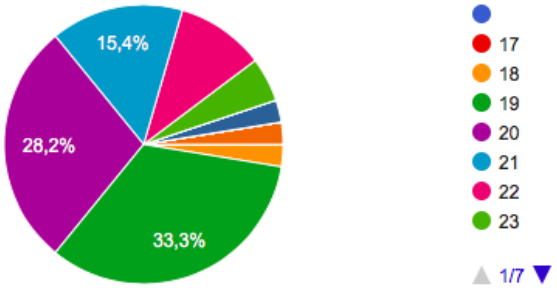


Figura 7.8: Edad.

7.3.1. De haber encontrado algún fallo ¿Podría escribir cual fue?

En esta pregunta 17 personas escribieron una respuesta. La gran mayoría (13 respuestas) comentaban que la descarga y la aplicación dio problemas, esto se debío al método de distribución de la aplicación que utilizamos, ya que al estar directamente como .apk y no de la tienda de aplicaciones *Android* algunos dispositivos solicitaban cambios en la seguridad del sistema.

Otro comentario a destacar dice que las respuestas a las preguntas desaparecen demasiado rapido, es decir los *Toast* para las respuestas correctas no están el tiempo suficiente para poder leerlas. Este probleme se solucionará en futuras versiones.

También otro comenta que tiene que introducir usuario y contraseña cada vez que tiene que utilizar la aplicación. Esto problema espera solucionarse para la siguiente versión, simplemente guardando la versión del usuario para que solo tenga que logearse una vez.

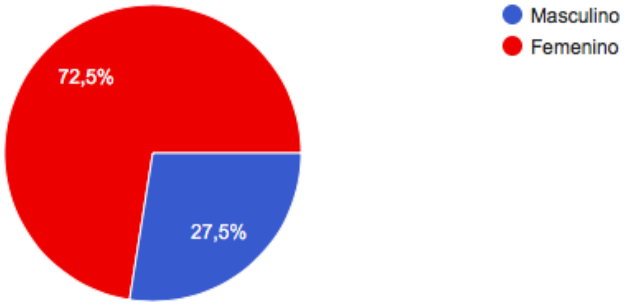


Figura 7.9: Sexo.

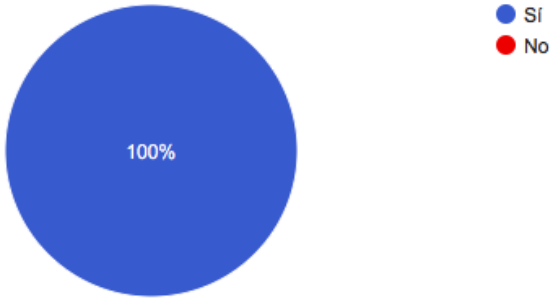


Figura 7.10: ¿Perteneces a la Universidad Autonoma de Madrid?.

7.3.2. Si pudiese mejorar algo ¿Qué sería?

Esta pregunta recibió 16 respuestas.

Seis respuestas comentaban acerca de la disponibilidad de la aplicación en todas la plataformas (iOS, Windows Phone). Con esto poco podemos hacer ya que se dicidió solamente hacer la aplicación para dispositivos Android.

Otros seis usuarios comentaron que el diseño o estética de la aplicación era básico y debería mejorarse y que podrían añadirse fotografias o ilustraciones para ayudar a tener una mejor interpretación de lo que se esta estudiando.

Una opinión a destacar habla de introducir la funcionalidad de ordenarlo por autores/generos para facilitar las busquedas.

Las tres opiniones restantes hablaban de problemas que hemos comentado anteriormente, como poder descargarla directamente de la *PlayStore* o poder hacer *login* automáticamente.

7.3.3. ¿Añadiría alguna mejora a la aplicación?

Once usuarios respondieron a esta pregunta.

Cinco usuarios comentan que la aplicación no necesita mejoras o que no sabrían que mejorar.

Otros como en respuestas a preguntas anterior, comentan la disponibilidad en dispositivos iOS, *login* automático y la posibilidad de fotos para hacerla más atractiva visualmente.

Hay dos respuestas a destacar, una de ellas propone la posibilidad de obtener una puntución en base a la respuestas correctas al final de cada ejercicio y por otro lado una herramienta de busqueda para acceder al contenido de forma más precisa.

7.3.4. Aparte de los ya propuestos (Fill the Gaps, True False y Multiple Choice) ¿Qué otro tipo de preguntas o actividades añadirías a una aplicación de este tipo?

Esta es la pregunta que más respuestas ha obtenido, un total de 23.

Aquí nos proponen otros tipos de preguntas para la aplicación, pero remarcar que eran ellos los que nos pasaban la información y nosotros solo nos encargabamos de volcarla en la aplicación y estabamos limitados a unicamente los tres tipos de preguntas, *Fill the Gaps, True False* y *Multiple Choice*, aunque estas proposiciones nos sirven como ideas para crear nuevos tipos de ejercicios para futuras versiones.

Entre los diferentes tipos que nos proponen, podemos marcar como interesantes, identificación de fragmentos de textos, respuestas de desarrollar, crucigramas, sopas de letras, unir preguntas con respuestas.

7.4. Explicación y valoración de los resultados totales

Estos resultados podemos valorarlos como positivos, la media de las notas de puntuación de las preguntas es 3.5/5. Aunque todavía con modificaciones por hacer y mejoras por añadir, podemos pensar que tanto los alumnos como la Dra. Ana González-Rivas Fernández a los que iba dirigida la aplicación han quedado satisfechos con el proyecto realizado y todas las sugerencias, opiniones y consejos que han realizado han sido valorados para tener una opinión por parte del cliente y así poder acercarnos más a la necesidad de estos para mejorar la aplicación en próximas versiones.

Capítulo 8

Conclusiones y trabajo futuro

Al fin y al cabo, somos lo que hacemos
para cambiar lo que somos.
Eduardo Galeano

8.1. Conclusiones del trabajo realizado

Partiendo del objetivo principal que era la creación de una plataforma móvil para la enseñanza de literatura inglesa, se ha buscado maximizar tanto la usabilidad como el diseño o la eficiencia de recursos durante el desarrollo. En está linea se han tomado diversas decisiones, entre ellas la creación de la aplicación unicamente para dispositivos Android y la plataforma Android Studio para su desarrollo, la elección de una base de datos remota mediante un servicio de terceros y la creación de una plataforma Web para facilitar a los estudiantes a incluir sus propios test en la aplicación.

Al finalizar el proyecto la aplicación era plenamente funcional y se puede acceder a su descarga. Pese que ha sido un proyecto para la Universidad Autónoma de Madrid, esta aplicación puede ser utilizada por todo el mundo que desee aprender sobre la literatura inglesa y repasar sus contenido y que el modelo también puede ser exportado a otras universidad, sin más que modificar la información almacenada y si se desea también a otras carreras para realizar en proceso de aprendizaje diferente y posiblemente más atractivo para los estudiantes.

Vamos a comparar los objetivos obtenidos con los requisitos planteadso inicialmente para poder extraer conclusiones sobre el trabajo realizado. Por ello vamos a recordar los requisitos del proyecto y extraer un grado de exito para cada uno de ellos.

- Realización de una aplicación que permita el estudio de diversos autores o epocas relacionadas con la literatura inglesa utilizando diversos

tipos de preguntas: Este requisito, que puede ser considerado el objetivo fundamental del proyencto, se ha cumplido exitosamente realizando una aplicación móvil Android, con lenguaje de programación *Java* utilizando las últimas librerias y guias de diseño propuestas por *Google* para hacerla intuitiva y agradable para el usuario.

- Creación de una plataforma Web que da la posibilidad a los alumnos de incluir sus propios test en la aplicación: Este objetivo podemos darlos por cumplido y para ello se ha realizado una página Web usando HTML, CSS y PHP, se han usado también bases datos MySql como elemento comunicador entre la plataforma Web y la aplicación. El uso de esta plataforma por parte del usuario no es del todo seguro ya que esta debe tener un buen sistema de verifiación para evitar problemas de introducción de datos en la base de datos.

8.2. Conclusiones personales

A lo largo de la realización de este proyecto hemos encontrado diversos problemas que han tenido que ser resultas. Tras haber definido los objetivos que teniamos que implementar y la forma en la que estos iban a ser resueltos he tenido que formarme en algunos entornos ya a pesar de conocer Java he tenido que aprender programación para orientada a dispositivos Android, que usaba este lenguaje pero se basaba en conceptos y librerias que personamente desconocia. Pero gracias a la formación académica durante los años de carrera ha sido lo suficientemente buena como para enfrentarme a los diversos problemas sin demasiado esfuerzo.

Quiero remarcar lo mucho que he aprendido durante todo el proyecto, sobretodo en la gestión y realización de proyectos, a pesar de que era un proyecto unipersonal, desde el principio de este con la evaluación de requisitos, objetivos, planificación con usuarios reales, diseño de la aplicacíon y de la base de datos y me ha ayudado a entender el punto de vista de estos y sus necesidades. Por lo que considero que la realización de este proyecto ha sido muy enriquecedora ya que he adquirido y puesto en práctica conocimientos y aptitudes fundamentales para el desarrollo de proyectos la cuales muy demandadas en la actualidad.

8.3. Trabajo futuro

Dado que este trabajo está limitado a la longitud propia de un TFG, las lineas de trabajo futuro que se pueden desarrollar son amplias. Una de las caracteristicas de este tipo de plataformas para realizar actividades es que normalmente son de uso único por la carrera a la que esta dedicado, en nuestro caso literatura inglesa pero prodria orientarse a todo tipo de carreras,

porque los contenidos para cada una son diferentes. Por ello en caso de que se quiera utilizar la aplicación en otra carrera sería necesaria la colaboración del personal encargado para realizar las modificaciones que se requieren.

Como mejoras para la aplicación movil podemos destacar, algunas de estas han sido sacadas de los comentarios que nos han dado los usuarios:

- Introducción de imagenes en las descripciones de lo autores. Esto haría un diseño mucho más atractivo para el usuario.
- Login automático. Que cada vez un usuario, que ya se ha logeado por primera vez, no tenga que hacerlo.
- Mejoras en el diseño. Cambiar el aspecto visual de las preguntas, añadiendo elementos que mejoren la experiencia del usuario.
- Sistema de calificación de preguntas. Cuando un usuario termine de responder todas las preguntas, el sistema muestre los resultados obtenidos por él.
- Sistema de búsqueda para los autores y test. La posibilidad de buscar entre los diferentes autores o test
- Sistema de seguimiento de alumnos. Que el profesor encargado pueda recibir información de los progresos realizados por los usuarios.
- Multiplataforma Desarrollo de la aplicaciones para sistemas operativos como *iOS* o *Windows Phone*

Para la parte de la plataforma Web, la mejora en la verificación de los questionarios enviados por los usuarios. Cambios en diseño de la página Web y la posibilidad de poder enviarlos con otros formatos y esctructuras.

Capítulo 9

Conclusions and future work

9.1. Conclusions of the work done

Having in mind that the principal objective was to create a mobile platform for the teaching of English literature, the usability, the design and the efficiency of resources have been maximized during the development. In this same line, diverse decisions have been made, such as the unique creation of the application for Androids and the platform Android Studio for its development, the election of a remote data base through a mediator service, and the creation of a web platform in order to facilitate the inclusion of tests in the application to students.

Once the project was finished, the application was totally functional and can be downloaded. Even though it has been a project for the Universidad Autónoma de Madrid, this application can be used by everybody who would like to learn about English literature and revise its content. This model can also be exported to other universities, with a simple touch: changing the accumulated information, as well as to other degrees to achieve a different process of learning, and possibly much more atractive for students.

We are now going to compare the objectives obtained with the requirements layed out at the beginning, in order to be able to extract conclusions about the work done. This is why we are going to remember the requisites of this project and to extract a success degree for each of them.

- The development of an application that permits the study of diverse authors and periods related with English literature, using diverse questions. This requirement, that can be considered the main and fundamental objective of the project, has been accomplished successfully

with the attainment of an Android mobile application, with a programming language Java, using the last libraries and design guides proposed by Google to make it intuitive and pleasant for the user.

- The creation of a web platform which gives the possibility to students to include their own tests in this application: This objective can been known as accomplished and for it a web page has been done using HTML, CSS and PHP. Data bases such as MySql have been used as a communication element in between the application and the web platform. The use of this platform by the user isn?t totally secure because it has to have a good verification system to avoid problems when introducing information in the data base.

9.2. Personal Conclusion

Throughout the development of this project we have come across diverse problems which have had to be solved. After having our objectives clearly defined and the way in which they were going to be solved, I have had to train myself in some aspects. Even though I knew how to use Java, I have had to learn programming to orientate it to Android devices. I used this programming language but was based in concepts and libraries which I personally didn't know. Nevertheless, my academic training over the years has been so good that I have been able to confront the problems without much effort.

I would like to emphasise how much I have learned during this project, most of all in the management and development of the projects. Since the beginning, even though it was a unipersonal project, the evaluation of requirements, objectives, planning with real users, and designing the application and the data base, has made me understand the point of view of the users and their necessities. I consider that the realization of this project has been rewarding because I have acquired knowledge and fundamental aptitudes that are required today for the development of such projects.

9.3. Future work

Considering that this project is limited to the length of TFG, the lines of future work can be developed are spacious. One of the characteristics of this type of platforms for activities is that they can be use by the degree that they are dedicated, in our case English literature but could face all kinds of degrees, because the content for each one are different. In case that you want to the mobile platform in another degree, it would be needed the collaboration of the developers to make the changes and modifications that are required.

As improvements to the mobile application we can highlight, some of these improvements have been taken from the comments that users have given to us:

- Introduction of images in the descriptions of the authors. This would make it a much more attractive design for the users.
- Automatic login. Each time a user, it is has logged in for the first time, does not have to do so.
- Improvements in the design. Change de visual appearance of the questions, by adding elements that improve the user experience.
- Answers rating system. When the user finishes all the questions ponder todas las preguntas, the system shows the results obtained by it.
- Search system for authors and test. The possibility to look for between different authors or test.
- Student tracking system. The responsible teacher can get information of the progress made by the users.
- Multiplataforma Development of applications for operating systems like iOS and windows phone.

For the web platform, the improvement of the questionnarie sent by the users. Changes in design of the website and the possibility to send questionnaries with other formats and strutures.

Bibliografía

Duolingo - idiomas gratis (2015).
https://play.google.com/store/apps/details?id=com.duolingo.

Aprender idiomas con busuu (2015)
https://play.google.com/store/apps/details?id=com.busuu.android.enc

Joan ribas Lequerica (2015). Desarrollo de aplicaciones para Android. Edicion 2016. Editorial Anaya.

Android Studio (2015)
https://developer.android.com/studio/index.html

Android Developers Page (2015).
https://developer.android.com/develop/index.html.

Material Design. Google design guide (2016)
https://material.google.com/.

Up and running with material design (2016).
https://developer.android.com/design/index.html.

Transmitting Network Data Using Volley (2016).
https://developer.android.com/training/volley/index.html.

Creating Lists and Cards. RecyclerView. (2016)
https://developer.android.com/training/material/lists-cards.html.

Crear Un Web Service Para Android Con Mysql, Php y Json (2015).
http://www.hermosaprogramacion.com/2015/05/crear-un-webservice-para-android-con-mysql-php-y-json/.